University of Crete

Computer Science Department

Design and development of a platform for the management and collaborative identification of co-reference on digital resources

Valantis Melesanakis

Master's Thesis

Heraklion, January 2011

ΠΑΝΕΠΙΣΤΗΜΙΟ ΚΡΗΤΗΣ

ΣΧΟΛΗ ΘΕΤΙΚΩΝ ΚΑΙ ΤΕΧΝΟΛΟΓΙΚΩΝ ΕΠΙΣΤΗΜΩΝ

ΤΜΗΜΑ ΕΠΙΣΤΗΜΗΣ ΥΠΟΛΟΓΙΣΤΩΝ

Σχεδιασμός και ανάπτυξη πλατφόρμας διαχείρισης και συνεργατικής ταύτισης πολλαπλών αναφορών σε ψηφιακούς πόρους.

Εργασία που υποβλήθηκε από τον

Βαλάντη Σπυρ. Μελεσανάκη

ως μερική εκπλήρωση των απαιτήσεων για την απόκτηση

ΜΕΤΑΠΤΥΧΙΑΚΟΥ ΔΙΠΛΩΜΑΤΟΣ ΕΙΔΙΚΕΥΣΗΣ

Συγγραφέας:

_____

Βαλάντης Μελεσανάκης, Τμήμα Επιστήμης Υπολογιστών

Εισηγητική Επιτροπή

_____

Γρηγόρης Αντωνίου, Καθηγητής, Επόπτης

_____

Martin Doerr, Ερευνητής, Μέλος

_____

Δημήτρης Πλεξουσάκης, Καθηγητής, Μέλος

Δεκτή:

_____

Άγγελος Μπίλας, Αναπληρωτής Καθηγητής

Πρόεδρος Επιτροπής Μεταπτυχιακών Σπουδών

Ηράκλειο, Ιανουάριος 2011

Design and development of a platform for the management and collaborative identification of co-reference on digital resources

Valantis Melesanakis

Master's Thesis

Computer Science Department, University Of Crete

Abstract

The historical analysis in its first step is based on finding associated documents. The most important kind of relationships that exist between the documents is the reference of the same entities independently of their name and description. In the digital era we live in, there are numerous digital documents available in the Internet and for that reason became one of the most important tools for the researchers. In this dynamic environment, it is useful to develop an information system which allows the researchers to manage the related documents, to find descriptions for the named entities and then to determine the identity of them based on their knowledge. A lot of attempts based on automatic named entity recognition (NER) have been done over the last years. Nevertheless, the success rates of NER are limited- it works well only in rich texts and it demands enough complete and comparable data. Moreover, many systems aim to classify or categorize documents which refer a specific entity. Furthermore few of the systems use the technologies of the semantic web in order to represent the user's description and identity assumptions. However, few of the systems provide enough support in order to help the users making their identity assumptions manually.

In this thesis we developed a model which allows in multiple users to express their own descriptions for the named entities. In this model it is possible to express distinctions of different knowledge by the researchers for named entities and their identity assumptions. Moreover it enables in user groups to operate collaboratively in order to express their different identity assumptions for persons and places and to correlate them with certain references in documents. We present the

“WhoWhereWhen”, an information system which uses the ideas of WEB 2.0 and enables user groups to work collaboratively. In this system the challenge was to develop a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of document citations and associated identity assumptions and to manage the collaborative process of manual entity description and distinction in different citations.

Supervisor: Grigoris Antoniou

Professor

Σχεδιασμός και ανάπτυξη πλατφόρμας διαχείρισης και συνεργατικής ταύτισης πολλαπλών αναφορών σε ψηφιακούς πόρους.

Βαλάντης Μελεσανάκης

Μεταπτυχιακή εργασία

Τμήμα Επιστήμης Υπολογιστών, Πανεπιστήμιο Κρήτης

Περίληψη

Η ιστορική ανάλυση στα πρώτα της βήματα βασίζεται στην εύρεση συσχετιζόμενων εγγράφων. Το πιο σημαντικό είδος συσχέτισης που υπάρχει μεταξύ των εγγράφων είναι η αναφορά ίδιων οντοτήτων σε αυτά τα έγγραφα ανεξάρτητα από το όνομα τους και την περιγραφή τους. Στην ψηφιακή εποχή που διανύουμε, υπάρχει πληθώρα διαθέσιμων ψηφιακών εγγράφων στο διαδίκτυο και για το λόγο αυτό, το τελευταίο αποτέλεσε ένα από τα σημαντικότερα εργαλεία των ερευνητών. Σε αυτό το δυναμικό περιβάλλον είναι χρήσιμη η δημιουργία ενός πληροφοριακού συστήματος το οποίο θα επιτρέπει στους ερευνητές να διαχειρίζονται τα συσχετιζόμενα έγγραφα, να εισάγουν τις περιγραφές τους για τις επώνυμες οντότητες που αναφέρονται σε αυτά και στη συνέχεια να οδηγούνται στην ταυτοποίησή τους βασιζόμενη στην δική τους γνώση για αυτές. Τα τελευταία χρόνια έχουν γίνει πολλές προσπάθειες που βασίζονται στην αυτόματη αναγνώριση των επώνυμων οντοτήτων οι οποίες όμως δεν έχουν επιτύχει υψηλά ποσοστά αναγνώρισης, περιορίζονται κυρίως στην επεξεργασία κειμένων και απαιτούν αρκετά, ολοκληρωμένα και συγκρίσιμα δεδομένα. Επιπρόσθετα, πολλά συστήματα επιδιώκουν να κάνουν ομαδοποίηση ή κατηγοριοποίηση των εγγράφων που αναφέρουν μια συγκεκριμένη οντότητα. Επίσης, ελάχιστα συστήματα χρησιμοποιούν τεχνολογίες του σημασιολογικού ιστού προκειμένου να εκφράσουν τις περιγραφές των χρηστών αλλά και τις υποθέσεις τους για ταυτοποίηση των οντοτήτων και επιπλέον ελάχιστα από τα συστήματα παρέχουν στους ερευνητές αρκετά βοηθητικά στοιχεία των διαφορετικών οντοτήτων προκειμένου να τους βοηθήσουν στην ταυτοποίηση αυτών.

Σε αυτή την εργασία αναπτύξαμε ένα πλήρες μοντέλο το οποίο επιτρέπει σε πολλούς χρήστες να μπορούν να εκφράσουν τη δική τους γνώση για τις επώνυμες οντότητες. Σημαντικό σε αυτό το μοντέλο είναι η διάκριση της διαφορετικής γνώσης που διαθέτουν οι ερευνητές καθώς εκφράζουν διαφορετικές περιγραφές για τις επώνυμες οντότητες καθώς επίσης και τα βοηθητικά στοιχεία ταυτοποίησης αυτών. Επιπλέον το μοντέλο επιτρέπει σε ομάδες χρηστών να λειτουργούν συνεργατικά προκειμένου να ταυτοποιούν επώνυμες οντότητες όπως ανθρώπους και τοποθεσίες και να τις συσχετίζουν με συγκεκριμένες αναφορές στα έγγραφα. Παρουσιάζουμε το WhoWhereWhen, ένα πληροφοριακό σύστημα το οποίο χρησιμοποιεί ιδέες του WEB 2.0 και επιτρέπει σε ομάδες ερευνητών να λειτουργούν συνεργατικά. Σημαντικό σε αυτό το σύστημα ήταν η ανάπτυξη μιας εύχρηστης και διαισθητικής διεπαφής χρήστη η οποία καθοδηγεί το χρήστη να διακρίνει τις έννοιες στις αναφορές των εγγράφων για τις επώνυμες οντότητες καθώς επίσης και τις συναφείς παραδοχές ταυτοποίησης. Επιπλέον επιτρέπει να διαχειρίζεται τη συνεργατική διαδικασία της περιγραφής των οντοτήτων και του διαχωρισμού των διαφορετικών αναφορών που πραγματοποιούν οι χρήστες.

Επόπτης Καθηγητής: Γρηγόρης Αντωνίου

Καθηγητής

.

Ευχαριστίες

Στο σημείο αυτό νοιώθω την ανάγκη να ευχαριστήσω τον επόπτη της εργασίας μου καθηγητή κ. Γρηγόρη Αντωνίου για την εμπιστοσύνη που μου έδειξε καθώς επίσης για τη στήριξη που μου παρείχε κατά τη διάρκεια της εργασίας μου. Επίσης θέλω να ευχαριστήσω θερμά τον δεύτερο επόπτη της μεταπτυχιακής εργασίας μου, ερευνητή κ.**Martin Doerr** για την άψογη και αποδοτική συνεργασία που είχαμε. Ήταν σημαντικό για μένα που μου έδωσε τη δυνατότητα να ασχοληθώ με ένα πολύ ενδιαφέρον και επίκαιρο θέμα. Θα ήθελα επίσης να πω ένα ευχαριστώ για τις εύστοχες και ουσιαστικές παρατηρήσεις του, για τη καθοδήγηση του αλλά και την προθυμία του κατά τη διάρκεια αυτής της εργασίας. Επίσης θέλω να ευχαριστήσω τον καθηγητή κ. Πλεξουσάκη για την προθυμία του να συμμετέχει στην εξεταστική επιτροπή.

Θα ήθελα επίσης να εκφράσω ότι νοιώθω πολύ τυχερός που βρέθηκα στο εργαστήριο πληροφοριακών συστημάτων ΙΠ/ΙΤΕ καθώς αποτέλεσε ιδανικό περιβάλλον εργασίας προκειμένου να ολοκληρώσω την μεταπτυχιακή μου εργασία. Επιπλέον τα μέλη του εργαστηρίου για εμένα εκτός από εξαιρετικοί συνεργάτες αποτέλεσαν και εξαιρετικοί φίλοι.

Ένα μεγάλο ευχαριστώ θα ήθελα να εκφράσω στους φίλους μου οι οποίοι ήταν πάντα δίπλα μου και με στήριζαν ο καθένας με το δικό του μοναδικό τρόπο. Θέλω όμως να ευχαριστήσω ιδιαίτερα την Ιωάννα που ήταν πάντα δίπλα μου σε κάθε εύκολη ή δύσκολη στιγμή και με στήριζε με όποιον τρόπο μπορούσε ακόμα και αν βρισκόταν πολλά μίλια μακριά. Η τεράστια υπομονή της αλλά και η ψυχολογική στήριξη της με βοήθησαν σημαντικά κατά τη διάρκεια της εργασίας μου.

Τελειώνοντας θα ήθελα να εκφράσω την ευγνωμοσύνη μου στους γονείς μου Βούλα και Σπύρο και την αδερφή μου Έφη για την υπομονή τους και για τη βοήθεια που μου παρείχαν κατά τη διάρκεια των σπουδών μου. Ήταν πάντα δίπλα μου και με στήριζαν σε κάθε βήμα της ζωής μου με το καλύτερο τρόπο.

Contents

1 Introduction 1
1.1 Introduction 1
1.2 The Problem 2
1.3 Contribution 4
1.4 Organization of this thesis 5
2 Related Work 7
2.1 Information systems, models and approaches for co-reference 7
2.1.1 Named Entity Recognition 7
2.1.1.1 Language Factor 8
2.1.1.2 Entity Type Factor 8
2.1.1.3 Categories of automatic NER and Extraction Systems 9
2.1.1.4 Evaluation 11
2.1.2 Co-reference Systems 11
2.1.2.1 GATE 12
2.1.2.2 LbjNerTagger 13
2.1.2.3 CRFClassifier 14
2.1.2.4 OOF 15
2.1.2.5 Consistent Reference Service (CRS) 16
2.1.3Authority Files - Knowledge Organization Systems (KOS) 19
2.1.4 Folksonomies 21
2.2 Summary and Discussion 24
3 Co-Reference and Identification Model 27
3.1 Introduction 27
3.2 Assumptions and basic terms 28
3.3 Web Documents 29
3.4 Areas on web documents 30
3.5 Citations 32
3.6 Co-reference 32
3.7 Identity Assumptions 33
3.8 Benefits using RDF vs. XML 37

4 WhoWhereWhen: A platform for the management and collaborative identification of co-reference on digital resources 39
4.1 Introduction 39
4.2 Assumptions and conditions 40
4.3 System description 42
4.3.1 System Architecture 43
4.4 System Platform and components 44
4.4.1 Storage Layer 44
4.4.1.1 RDF Sesame Repository 45
4.4.1.2 Query Language 46
4.4.1.3 File Repository 47
4.4.2 Repository Layer 47
4.4.2.1 Repository Manager 48
4.4.2.2 File Repository Manager 49
4.4.2.3 Caching 50
4.4.2.4 Normalization 51
4.4.2.5 Document downloading 53
4.4.3 Behavioral Layer 55
4.4.3.1 Document Downloader 55
4.4.3.1.1 Image Downloader 56
4.4.3.1.2 PDF Downloader 57
4.4.3.1.3 Page Downloader 57
4.4.3.2 Search Components 58
4.4.3.2.1 Entity Search 58
4.4.3.2.2 Search Document which cites named entities 59
4.4.3.2.3 Citation Search 60
4.4.3.2.4 Identity Assumption Search 61
4.4.3.3 Knowledge Submitter 61
4.4.3.3.1 Import of Named Entities 62
4.4.3.3.2 Citations 63
4.4.3.3.3 Identity Assumptions 64
4.4.3.3.4 Split 64
4.4.3.3.5 Instance URIs and Contexts 65
4.3.2 Data Flow 66
4.4.4 Representation Layer 67
4.4.4.1 Documents 68

4.4.4.1.1 Images 68
4.4.4.1.2 PDF files 69
4.4.4.1.3 HTML pages 71
4.4.4.2 Example of use 75
4.4.4.3 User Interface – Document Manager 80
4.4.4.3.1 Document Search 83
4.4.4.3.2 Document View 84
4.4.4.3.3 Make Citation 86
4.4.4.3.4 Entity View 90
4.4.4.3.5 Entity Search 91
4.4.4.3.6 Same As editor 93
4.4.4.4 Programming Languages and techniques that were used on the WWW platform 95
5 Conclusions and future work 97
Appendix A.1 RDF Source 105

List of Tables

List of Figures

Figure 2. 1 Screenshot of GATE 13
Figure 2. 2 Screenshot of LBJ Tagger 14
Figure 2. 3 Screenshot of CRFClassifier 15
Figure 2. 4 A screenshot of Open Ontology Forge which shows the co-reference annotation procedure. 16
Figure 2. 5 A screenshot of Open Ontology Forge which shows the co-reference annotation procedure with images. 16
Figure 2.6 A mockup of CRS interface 17
Figure 2.7 Example Bundle Graph 19
Figure 2. 8 Search page of VIAF 21
Figure 2. 9 Flickr Interface 22
Figure 2. 10 Flickr interface 23
Figure 2. 11 Web site of Cleveland Museum of Art 23

Figure 3. 1 Hierarchy of Named Entity 29
Figure 3. 2 Types of Infromation documents 30
Figure 3. 3 Types of document Areas 31
Figure 3. 4 Types of documents, areas and subroperties of Included in property 31
Figure 3. 5 The citation which determine a Document Area and cites a named Entity 32
Figure 3. 6 Co-reference representation 33
Figure 3. 7 Types of identification 34
Figure 3. 8 Person Identification 35
Figure 3. 9 Place Identification 36
Figure 3. 10 Example of Identity assumption 37

Figure 4. 1 WhoWhereWhen Platform Architecture 42
Figure 4. 2 The flowchart of Document downloader 56
Figure 4. 3 Data flow of WhoWhereWhen 66
Figure 4. 4 Determination of Image Area 69
Figure 4. 5 WWW PDF viewer 71
Figure 4. 6 Example of selection 72
Figure 4. 7 Selection Example 73
Figure 4. 8 The named entities and documents which supported by user Gregory 76
Figure 4. 9 The named entities and documents which supported by user Martin 77
Figure 4. 10 Determination of Areas in Document Image 78
Figure 4. 11 The named entity and document which supported by user Valantis 78
Figure 4. 12 Identity Assumptions of users 79
Figure 4. 13 User model of WWW 82
Figure 4. 14 Login page of WhoWhereWhen 83
Figure 4. 15 Document Search 84

Figure 4. 16 Document View in images 85
Figure 4. 17 Document View in PDF files 85
Figure 4. 18 Document View in HTML pages 86
Figure 4. 19 Area determination on the document 87
Figure 4. 20 The user enters his knowledge on KB 88
Figure 4. 21 Citation search 89
Figure 4. 22 Co-reference determination 89
Figure 4. 23 Entity View 90
Figure 4. 24 Entity View 91
Figure 4. 25 Entity Search 92
Figure 4. 26 Entity Search 92
Figure 4. 27 The user Martin makes his own identity assumptions for two different named entities 93
Figure 4. 28 The users search for assumptions which is associates with this entity 94
Figure 4. 29 The graphical representation of identity assumptions. 95

Chapter 1

1 Introduction

1.1 Introduction

Nowadays, the Internet and Digital Libraries is one of the dominant knowledge resources and is becoming a more and more important tools for historical analysis. The first steps of historical analysis are based on finding related documents. The most important kind of relationships between documents is that they refer to the same entity, so-called co-reference. The reference of the same entity is independent from the name or the description we could find on the documents. Very common cases of co-reference are place names which appear in documents with different names or description of them, also persons who appear with nicknames or different persons with the same names or images that show the same person. It is important for users to determine the co-reference of a named entity in documents and to express their supporting attributes for the named entities in order to help other users to make identity assumptions about them. The ideas of WEB 2.0 are a significant motivation because it is the ideal environment for users to find and share related documents with others. In addition, it is important for the users to interact with documents in order to determine the specific areas that cited the named entities. On the other hand, technologies of semantic web can provide important support in order to express human knowledge for named entities and identity assumptions about them.

Most of the information systems that have been developed so far mainly aim to recognize named entities automatically, are limited only to text documents. In most cases, they deal with the classification of named entities rather than the co-reference of named entities in different documents. However, the automatic recognition has limited success rates and demands comparable, complete and enough data. In this case

the human knowledge is superior. The information systems that recognize text have limited operations and ignore the Internet as an interaction and collaborative platform which provides to the users different types of documents. Those systems which use the Internet provide limited support to users for named entities and as a result they do not help them to express their knowledge for a named entity and to make their identity assumptions for different named entities. Furthermore, most of the information systems do not determine the identity of named entities but aim simply to make the classification of similar named entities or to represent the co-reference of this entity within documents.

In this thesis we aim to develop a model which describes the knowledge of users about the named entities such as persons and identity assumptions of them. The challenge in this model is the distinction of the different knowledge that users express for their descriptions and identifications. It is important to enable the users to express their different perceptions about the identity of named entities which are referred in documents. The most important part of this thesis was the development of an information system which helps the users to manage related documents, to associate the citation on them with the entities and to provide support in order to help other users to make their own identity assumptions. It also supports citations in multiple types of documents such as images, PDF documents and HTML pages. Due to the difficulty to show the different knowledge of users it was important to develop a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of documents citations for named entities and associated identity assumptions.

1.2 The Problem

In the digital era we live in, the Internet is becoming one of the main tools for historical analysis. The most important reason of the rapid growth of Internet was the large amount of available data. In this dynamic environment the users can share data resources in a variety of different types such as images, PDF files and HTML pages. The Internet became today a collaborative platform in which the users can easily search and find related documents that have reference to the same entity (such as persons and places), so-called co-reference. It is important in this environment that the

users manage the related documents, express their descriptions for the named entities and then make their identity assumptions for them.

There are a lot of information systems which aim to recognize automatically references to named entities in text documents. These systems are limited only to text documents and do not provide functionality in order to process documents of other types such as images, HTML pages and PDF files. Furthermore, they are used only by one user who loads the text document and do not allow to different users to work collaboratively. The Named Entity Recognition which is the main functionality that these information systems provide, has limited success rates because it is based on heuristic and grammatical rules which are used for the text process and aims to represent the reference of an entity and do not determine the identity of it. In addition, the Named Entity Recognition is a difficult process because it demands enough, complete and comparable data.

Nevertheless the CRS system (see 2.1.2.5) provides the bundle framework which enables to users to match manually documents that refer to a specific resource. In this system, the description of resources is limited and correlates the URLs of documents with them. In addition, it does not determine areas of the documents that refer to the specific resource and creates misinterpretations about the identity of resources.

Furthermore, other approaches such as Knowledge Organization Systems (KOS), link and match authority files of national libraries in order to determine the co-reference of entities on them. They associate the entities with a preferred unique identifier. Some KOS may aim to determine the co-reference of an entity between different authority files but do not link to references in documents. Finally, the folksonomies are produced by users who collaboratively tag content on web. This method is used only for the classification and categorization of the content and do not aim to identify named entities that are referred on this.

Summarizing, one of the problems which we aim to solve in this thesis is to develop a model which enable to different users to express their own knowledge about the named entities and identity assumptions of them. The challenge on this model is to determine the distinctions of knowledge that different users express about the descriptions and identity of named entities. In addition, it is important to develop an information system which supports this collaborative environment of users in order

to express their different knowledge about their descriptions and identity assumptions. Because of the difficulty to represent graphically the different knowledge of users, it is important to develop a user friendly and intuitive interface that guides the user to distinguish the citations in documents that other users have created and to study the associated identity assumptions. Finally, it is important to support annotations in different types of documents such as PDF, HTML pages and images.

1.3 Contribution

In this thesis we made a detailed survey of the related work of systems and approaches which deal with co-reference. More specifically we analyzed different domains and aspects of this topic such as name entity recognition approaches, systems for named entity recognitions, authority files and folksonomies in order to find elements which are useful to our work and problems or domains which are not supported by these existing solutions.

An important part of our work was the development of a model which enables a group of users to express their descriptions for named entities and the supporting identity information for them. In this collaborative community each user expresses his knowledge about how he saw the things on his mind. It is of major importance this model to support the distinctions of different knowledge that the users provide. In addition it enables the users to express their identity assumptions for named entities. Moreover it allows users to correlate the references of entities in a variety of documents.

In this thesis the significant contribution was the development of an information system based on the web which provides a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of documents citations for named entities and associated identity assumptions. This system enables the users to manage documents in the web which co-refer same entities and for that reason they are related to each other. The users can define the specific area on the document that a named entity is referred. Moreover the system helps and guides the users to study the other users' knowledge for named entities and their identities assumptions in order to help them express their descriptions. The Internet as an ideal

collaborative community provides documents in a variety of types. For this reason the WhoWhereWhen system enables users to manage documents of different types and specifically Images, PDF and HTML pages. In addition the users export their knowledge entries from the system, structural in RDF format in order to use it in other systems. Finally, we demonstrate an example based on real conditions in order to explain the usage and facilities of this system.

The developed system provides a complementary solution to NER and automated co-reference detection will be used in order to support the users to determine the co-reference of named entities on the documents.

1.4 Organization of this thesis

This thesis is organized as follows:

Chapter 2 describes the related work and presents other systems and approaches which are relative with the co-reference. Finally, it makes a comparison between the existing solutions and our work.

Chapter 3 analyzes the model that we developed in order to represent the knowledge of the users about the named entities and determine their identity assumption. Moreover, it determines how this model makes distinctions of the different knowledge of users.

Chapter 4 presents the WhoWhereWhen, a web based platform which enables users to make their own descriptions for named entities, determine citation on documents and finally to make their own identity assumptions for them. In addition, we describe the components and the architecture of this platform.

Chapter 5 contains the conclusion of this thesis and determines its future work.

The appendix which follows in the end of this thesis contains the model (chapter3) in RDF code.

Chapter 2

2 Related Work

2.1 Information systems, models and approaches for co-reference

There is a variety of applications and approaches which aim to represent the co-reference between documents or aim to recognize named entities automatically. Due to this, the related work of this thesis can be divided into four parts; in the first part, we investigate systems and approaches that aim to recognize Named Entities, mainly in text documents. In the next part, we refer to other co-reference tools. In view of these systems we will analyze the ways that references and citations are presented and how these systems recognize the named entities. The authority files are an important part of this investigation because they use duplicate detection processes in order to determine duplicate records which come from different sources. Finally, we refer to folksonomies, systems of classification derived from the collaborative creation and management of tags to annotate and categorize content.

2.1.1 Named Entity Recognition

Named Entity Recognition (NER) is a part of information extraction and includes the processing of structured and unstructured documents along with the determination of expressions that refer to persons, places, organizations and companies. In recent years, automatic named entity recognition has become one of the most popular research areas and a lot of approaches and methods have been developed.

NER involves two tasks [2], the first being the identification of proper entities mainly in text documents and the second the classification and presentation of these entities into a set of predefined categories of interest that a user has expressed.

For humans NER is a very simple process because many named entities start with capital letters and can be easily recognized, but for a machine this is too hard. Also, the main problem is the classification of this entity and the recognition of its type[4]. A very common example for this problem is the word *June* since it may have quite different meanings depending on the context. The machine cannot recognize if this word refers to a person or to the month June. Another example might be the criteria under which the machine recognizes the phrase *White House* as a location or as organization.

The NER has been mainly used for keyword extraction in text documents in a lot of spoken languages, analyzing text using grammar, symptomatic and heuristic rules. In the rest of this section the main approaches, systems and techniques for NER are analyzed [1].

2.1.1.1 Language Factor

Language is a very important factor for the Named Entity Recognition. The research is mainly devoted to English but the German language is also well studied [2]. Many other languages have been also studied like Spanish, Dutch, Chinese[5], French[6] ,Greek[6][7],Italian[8].Many other languages received some attention as well: Basque[9], Bulgarian[10], Danish[11] ,Korean[9] ,Romanian[12] , Russian [13], Swedish[14] and Turkish[15].

2.1.1.2 Entity Type Factor

Regarding this factor there are lot of practical or even philosophical problems. For instance, *the automotive company created by Henry Ford* in 1903 is referred to as Ford or Ford Motor Company. Rigid designators include proper names as well as certain natural kind terms like biological species and substances. Generally there are agreements within the NER community about the numerical expressions such as the amount of money and other types of units. While some instances of these types are

good examples of rigid designators (e.g., the year 2001 is the 2001st year of the Gregorian calendar) there are also many invalid ones (e.g., in June refers to the month of an undefined year – past June, this June, June 2020, etc.). It is arguable that the NER definition is loosened in such cases for practical reasons.

Early work formulates the NERC problem as recognizing "proper names" in general[16]. Overall, the most studied types are three specializations of "proper names": names of "persons", "locations" and "organizations". The type "location" can in turn be divided into multiple subtypes of "finegrained locations": city, state, country, etc.[17]. A recent interest in bioinformatics led to many studies being dedicated to such types as protein, DNA, RNA[18]

2.1.1.3 Categories of automatic NER and Extraction Systems

In recent years the automatic NER and Extraction systems have become very popular and a lot of important studies that have been addressed aim at developing these systems. These systems can be divided into three main categories[19]: Hand-Made Rule Based NER, Machine Learning-based NER and Hybrid NER.

In Hand Made Rule Based NER the systems focuses on extracting Entities using lots of human made rules sets .They are mainly based on a set of patterns using grammatical, syntactic and orthographic features (e.g. capitalization) and in many cases in combination with dictionaries[20]. Appelt et. al.[21][22] proposed a name identification system based on carefully handcrafted regular expressions called FASTUS. In this approach they divide the process into three steps: Recognizing Phrases, Recognizing Patterns and Merging incidents. Other approaches like Iwanska[23] use other external specialized resources such as gazetteers while Morgan [24] uses a highly sophisticated linguistic analysis.

These kinds of systems produce better results in restricted domains and applications but they have significant difficulty in detecting complex entities. Moreover the rule based systems suffer from the lack of portability and robustness and they have a high cost in maintaining the rule while at the same time they slightly change the data [2]. These systems can easily adapt to specific domains and languages but not necessarily well to new domains and languages.

In machine Learning Based NER the purpose of Named entity recognition is to convert identification problems into classification problems and then use a classification statistical model to solve them. In these approaches the system looks for patterns, nouns and relationships in the text mainly using learning algorithms. The next task is to identify nouns into classes such as persons and locations, using learning algorithms. We can define three main categories: Supervised Learning(SL), Semi-Supervised Learning(SSL) and Unsupervised Learning(UL).

- **Supervised Learning(SL)**

  In supervised learning, a program is used that can learn to classify a given set of labeled examples according to the same number of features. It is called supervised because the people mark up the training examples and are teaching the program towards the right directions. This approach requires a large amount of order for it to achieve a good performance.

- **Semi-Supervised Learning(SSL)**

  SSL Approach involves limited supervision such as a set of seeds, for starting the learning process. The system is, at first, trained to recognize entities in many contexts. Then the system tries to recognize these entities in other contexts. The learning process is then reapplied to newly found examples, so as to discover new relevant contexts. By repeating this process, a large number of names and a large number of contexts will eventually be gathered.

- **Unsupervised Learning(UL)**

  This is a different approach because the systems learn without any feedback. The goal in these systems is to make representations from data. The goal of the program is to build representations from data. These representations can then be used for data compression, classifying, decision making, and other purposes.

  Unlike rule based methods, these methods can easily conform to different domains or languages.

Finally, in Hybrid NER systems the approach uses a combination of rule-based and machine learning methods and tries to use the strongest points of each method. Although this type of approach can get better results than some other approaches, the weakness of handcraft Rule-base NER remains the same as that deriving from when there is a need to change the domain of data

2.1.1.4 Evaluation

The evaluation of NER systems is based upon the recognition of a named entity in a string and not in superficial appearance. In order to evaluate the results of NER systems, evaluation measures Precision(P) and Recall (R) borrowed from the information-retrieval community. That is:

$$\text{Precision} = \frac{number\ of\ correct\ rensponses}{number\ of\ rensponses}$$

$$\text{Recall} = \frac{number\ of\ correct\ rensponses}{number\ of\ relevant\ documents}$$

These two measures of performance combine to form one measure of performance, the *F*-measure, which is computed by the uniformly weighted harmonic mean of precision and recall:

$$\text{F-measure} = 2 * \frac{Recall * Precision}{Recall + Precision}$$

2.1.2 Co-reference Systems

The systems which were developed for co-reference mainly deal with linguistics. The most common case of co-reference appears in the presence of pronouns and anaphora. Anaphora is when a word is used to refer back to another word that occurred previously, such as *it* and *do* in “I know *it* and he *does* too”. In the sentence “I saw John today, he was jogging” the words “John” and “he” are most likely co-referent, as they probably refer to the same person[25]. This example may

seem simple for human abilities, but instructing a natural language processing system to do the same task is very difficult and has many complications and approaches. Moreover in these systems the entities recognized were classified in the appropriate classes.

There are a lot of systems which support many languages which, however, are not multilingual whereas other systems[33][34] are multilingual and can easily extend to new domains, tasks and languages.

Below we provide some characteristic and popular works:

2.1.2.1 GATE

General Architecture for Text Engineering (GATE)[26], developed by the University of Sheffield is nearly 15 years old. This system is an infrastructure for developing and deploying software components that process human language. It is an open source system which not only serves as a framework for text engineering but also is an architecture and a development environment in itself. ANNIE, A Nearly – New Information Extraction was deployed in GATE and provides functions such as NER. The main processing components for English Text provided by GATE are:

- Tokenizer
- Semantic tagger
- Orthomatcher
- Coreferencer

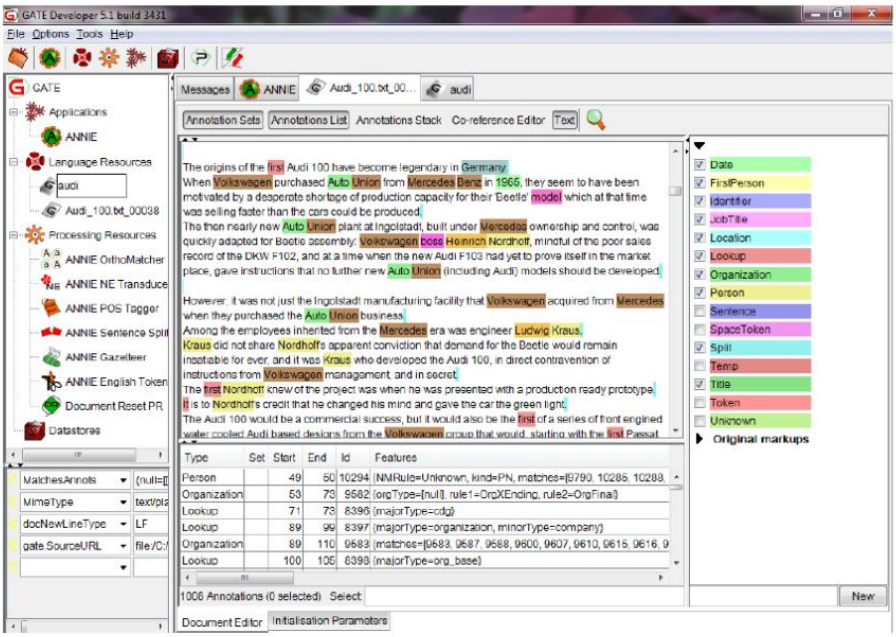


Figure 2. 1 Screenshot of GATE

The GATE system was developed in JAVA. ANNIE, also developed in JAVA, relies on finite state algorithms and the JAPE language which consists of a set of phrases, each of which consists of a set of patterns/action rules[27].

2.1.2.2 LbjNerTagger

This system[28] developed by the University of Illinois at Urbana-Champaign tags plain text with named entities (people / organizations / locations) and it uses gazetteers extracted from Wikipedia. It is based upon (Learning Based Java) LBJ[29][30]

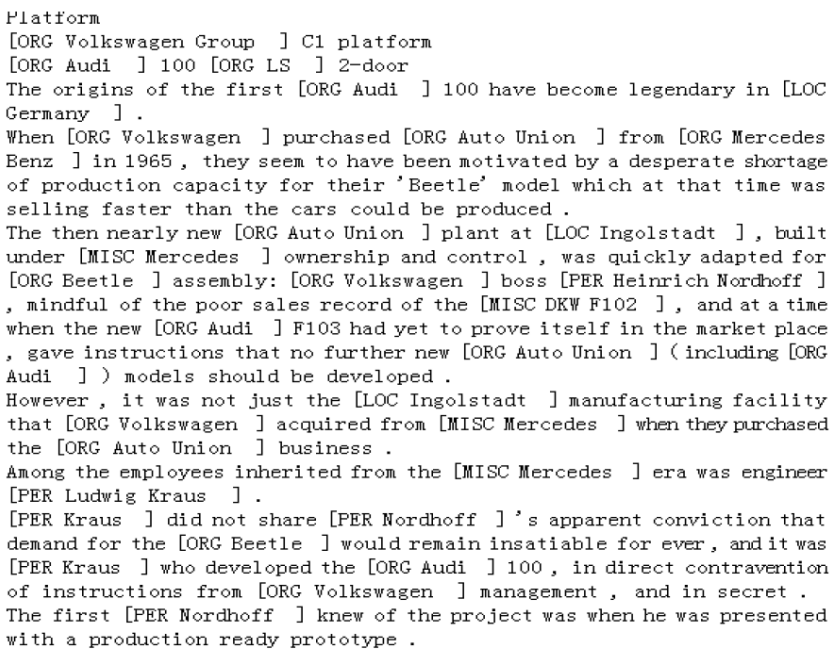


Figure 2. 2 Screenshot of LBJ Tagger

As we can see in the interface's mock up it includes Tag Sets: [PER person]; [LOC location]; [ORG orgnization]. The different tags relate with each other by a Learning Based Java program.

2.1.2.3 CRFClassifier

The CRFClassifier[31] is a named Entity recognition System developed in Java. It is developed by Jenny Finkel at the University of Stanford. The software provides a general (arbitrary order) implementation of linear chain Conditional Random Field (CRF) sequence models, coupled with well-engineered feature extractors for Named Entity Recognition.

The University of Stanford entered the 2003 CoNLL NER shared task, using a Character-based Maximum Entropy Markov Model (MEMM). In late 2003 they entered the BioCreative shared task, which aimed at doing NER in the domain of Biomedical papers which requires identifying genes and proteins, but not distinguishing between the two. Later in 2004 they entered the BioNLP shared task at CoLing which also looked at Biomedical papers, but required identifying five different classes - DNA, RNA, cell line, cell type, and protein.

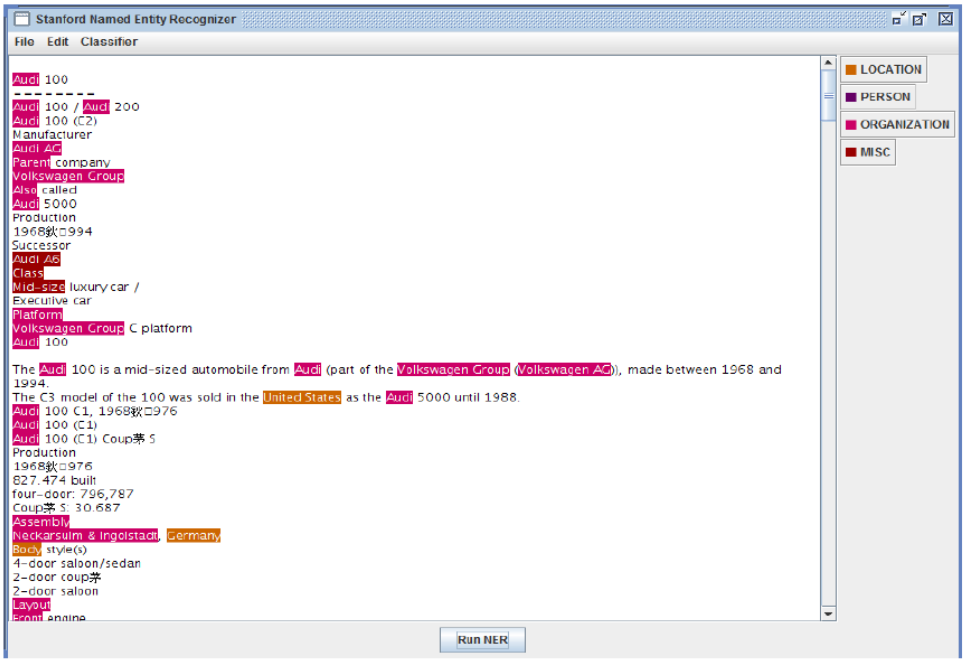


Figure 2. 3 Screenshot of CRFClassifier

2.1.2.4 OOF

In this subsection we will outline a system[32] which supports the annotation not only within linguistic expression but also within image data. It has applications in the biomedical domain in which images often contain important information for analyses and diagnoses, and the designers consider that linking images to textual descriptions of their semantic contents in terms of co-reference relations is useful for multimodal access to information. Moreover in this work they introduce a new called term named co-reference pool which is a set of linguistic annotations and photo annotations which refer to the same Entity. In figures 2.4 and 2.5 we can see the procedure of annotation in Linguistic Expressions and Images.

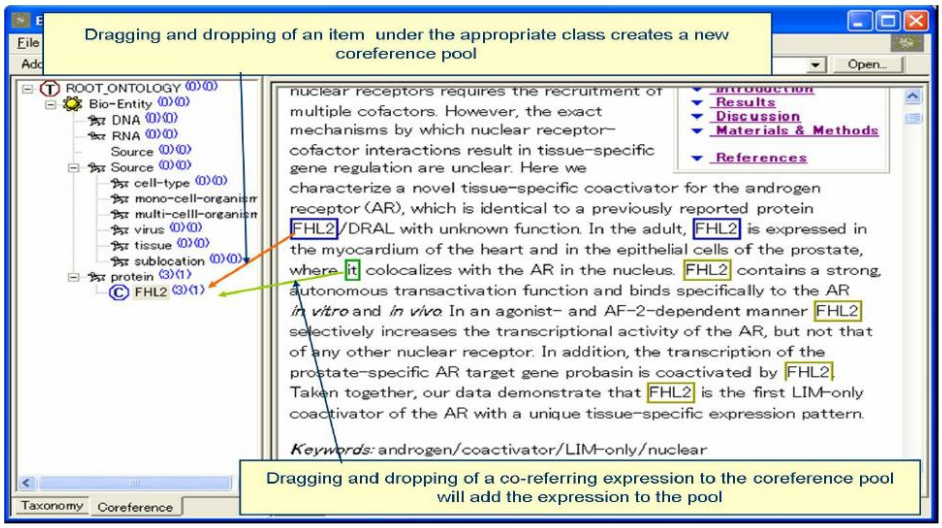


Figure 2. 4 A screenshot of Open Ontology Forge which shows the co-reference annotation procedure.

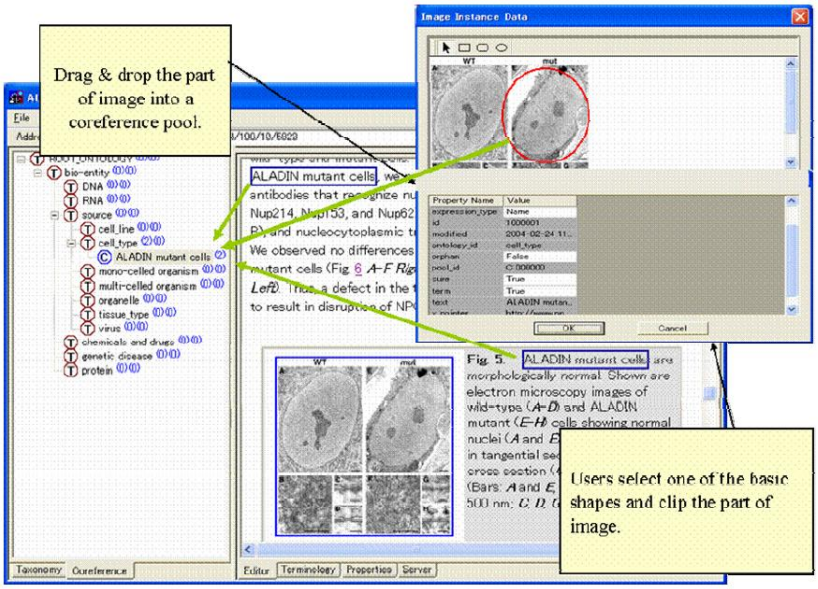


Figure 2. 5 A screenshot of Open Ontology Forge which shows the co-reference annotation procedure with images.

2.1.2.5 Consistent Reference Service (CRS)

There is an alternative system that not only deals with co-reference in linguistics but can be deployed in a range of applications. This system is called Consistent

Reference Service (CRS) [25] and has been developed by the University of Southampton. The CRS is designed to be a thesaurus-like reference that can be used by semantic applications as a source of co-reference resolution. An application may look up a reference for which it has information about and discovers other URIs that correspond to the same entity. The CRS achieves this by storing and making available established mappings so individual applications can become independent from the need to develop their own costly resolution systems. It is composed in two parts: a method for effectively representing co-reference and a communication mechanism. It includes 3 services:

- Import
- Export
- Update

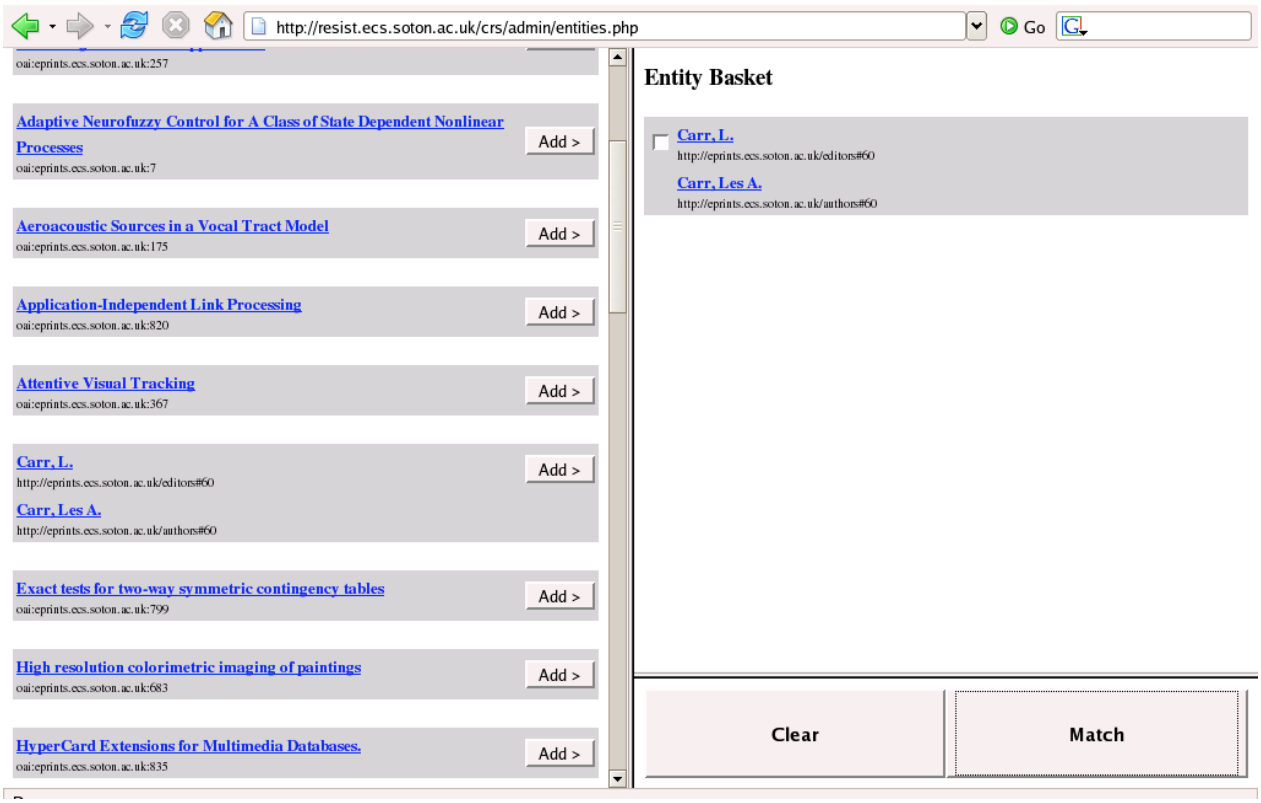


Figure 2.6 A mockup of CRS interface

In figure 2.6 we can see an interface's mockup of the CRS system. In the left part there are the bundles which provide metadata for the entity in order to help the user perform the matching. In the right part there is the entity basket in which the user adds the bundles from the left. After this process the user decides if those given bundles are equivalent or not.

The interface is used for performing keyword searches on literal values. The results are displayed as readout of the matching bundles and their contents, upon which a variety of operations can be performed. It is possible to merge bundles, delete bundles, and remove references from them. When a reference is removed, a new singleton bundle is created containing just the removed reference. If a bundle is deleted all the references are reset into singleton bundles.

The framework used in the CRS system is the Bundle framework. The designers of the Bundle framework try to annotate and merge instances of co-references with a more efficient and flexible manner than OWL.

The Bundle is a method for co-reference representation in order to merge references between systems. There are sets which contain equivalent and non-equivalent references to a resource. Bundles are metadata structures in RDF and OWL, and have the following features as described in [25]:

- Each bundle contains a set of references that are believed to refer to the same resource under a given set of circumstances. The predicate "hasEquivalentReference" is used to denote this.

- A bundle may contain a second set of references that explicitly do not refer to the same resource as the first set. This is achieved using "hasNonEquivalentReference" and does not imply anything about what the references do refer to; just that they do not represent the same resource as the bundle.

- There is an optional allowance for explicit context. A bundle is only said to be applicable within a specific context. If bundles from multiple contexts are stored together they may be differentiated by a BundleContext element. This is connected using the predicate "hasContext"

- A single reference in each bundle may be marked as canonical with the predicate "hasCanonicalReference". This is used to indicate a preferred reference to be used in new assertions regarding this entity, in this context. It is an optional addition but may be useful in consolidating the number of different URIs being used.

The example in fig 2.7 shows an RDF Graph for a Sample Bundle. In this example there is a bundle created to collate references for Hugh Glaser. As we can see two references are referred to him, one has been chosen as canonical. Moreover one reference to a "Henry Glaser" has been identified as not being the same person.

The diagram suggests a method of assigning URIs to bundles: the checksum of the combined URIs for the canonical reference and the context is appended to the base URI of the knowledge base it came from. This would always be unique when two bundles with the same URI, using this scheme, would become correctly merged.

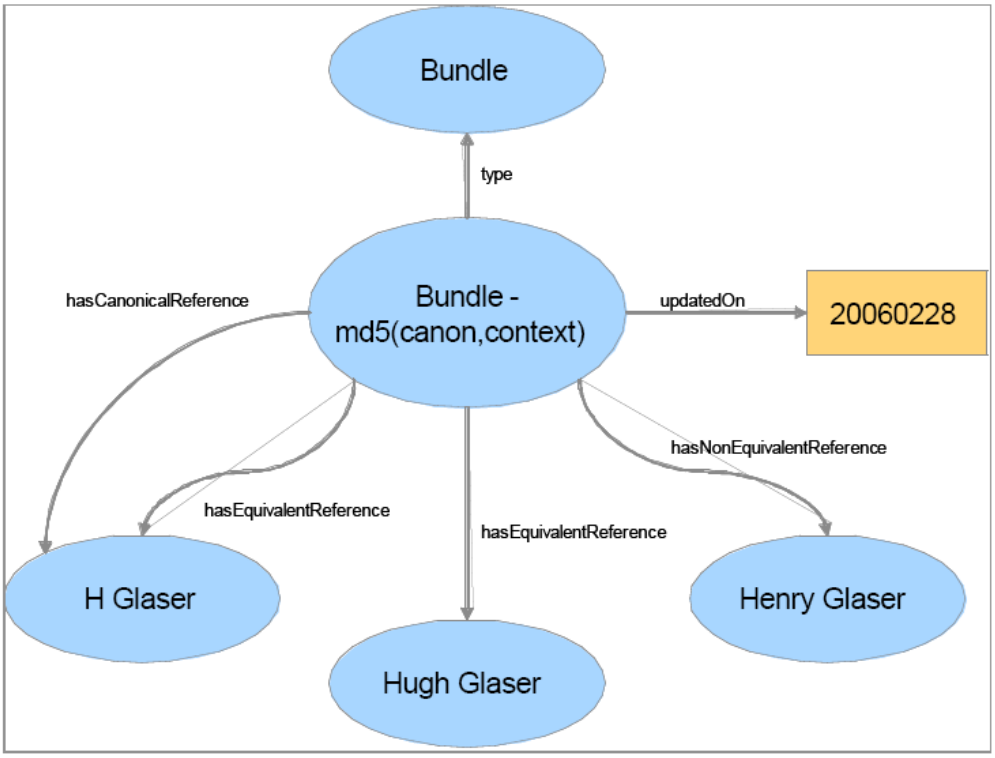


Figure 2.7 Example Bundle Graph

2.1.3Authority Files - Knowledge Organization Systems (KOS)

Most libraries and cultural heritage institutions has developed tools for organizing large collections of objects such as books or museum artifacts. In many cases these systems are accessible to users which use and extract their content. Many

institutions as they use different systems, they create vocabularies which are heterogeneous and maintain records in different ways. Moreover things are more complicated when the context is multilingual and many records are in different languages. In order to address this problem librarians, scholars and scientists have developed Knowledge Organization Systems (KOS) which match and link authority files of national libraries in order to determine references of specific items. The items recognized are associated with a preferred, unique representation in a central resource.

A user can match a description with another description under his/her own control only if the KOS can list enough properties of the described items. In addition he/she can use the preferred unique identifier that KOS associates with the items in order to define them in his/her source. This method can be applied if the preferred identifiers are globally unique, correctly applied and the KOS does not change the representation later. Moreover KOS is also used to support NER because it helps to decide if a word in a text might be the name of an entity [3].

The main KOS are VIAF [35], LEAF [36] for matching and linking authority records for personal names, MACS [37] for subject headings as the Library of Congress [38] implements specifying a range of subject headings that different literatures may use to describe their contents. These systems use Data mining methods (Named Entity Recognition) such as Name Match Confirmation of VIAF [35] which tries to identify names which were written in different ways due to the multilingual representation.

These systems fail when a person has pseudonyms or alternative names which are very different from the real name. In many cases like VIAF and LEAF they try to extract person names making use of relative dates like birth-date and/or death-date but these dates are available rarely and this leads to mismatches. Also VIAF uses bibliographic files in order to extract other properties regarding a person in order to make the case he/she may be the same within different sources[35].

Finally, Authority files try to collect the files which have citations for an entity which, though, do not provide essential properties for this entity in order to help the identification of it and the determination of a co-reference of it in multiple documents. In fig 2.8 we can see the search page of VIAF for the Traven[1] writer whose real name,

[1] http://en.wikipedia.org/wiki/B._Traven

nationality, date and place of birth and biographical details are all subject to dispute. In this example Traven has been identified from bibliographic files and we do not know essential properties of him in order to help us verify his identity.

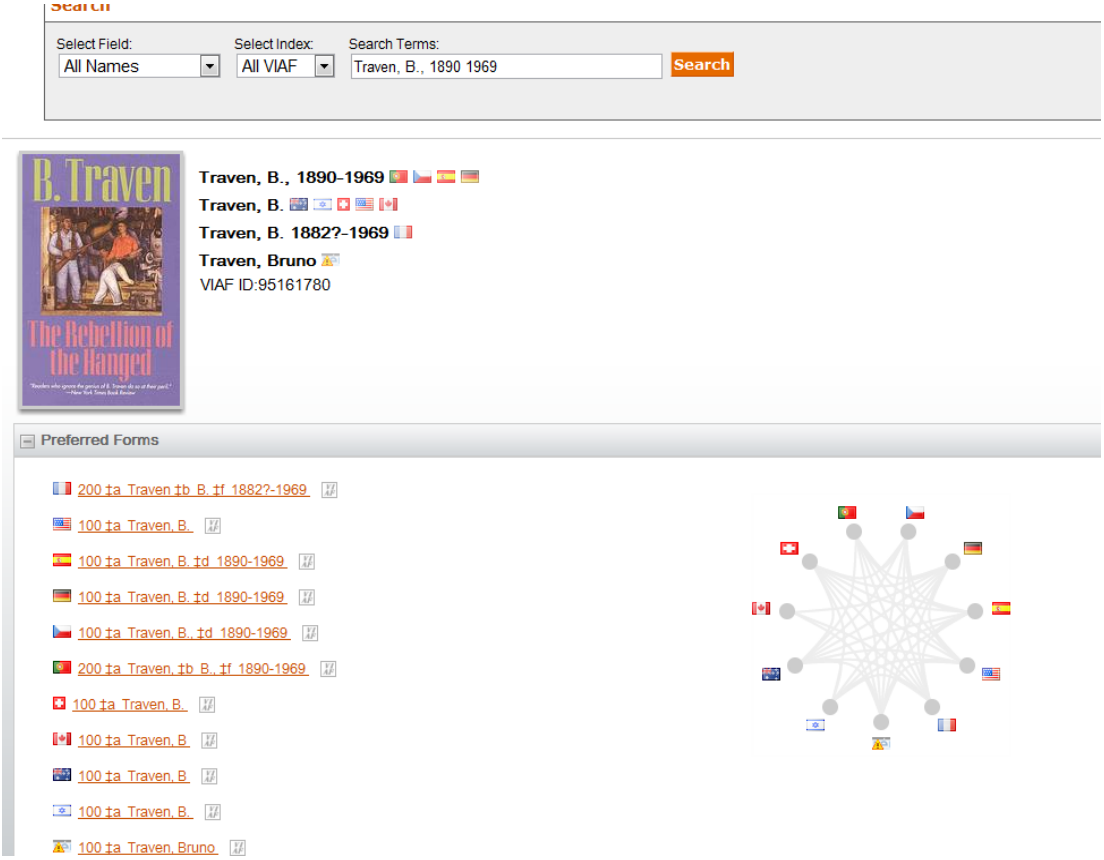


Figure 2. 8 Search page of VIAF

2.1.4 Folksonomies

Folksonomy[39] is a method for classification and categorization. It is based on collaborative creation and management of tags which users create in order to annotate and categorize content. This practice is also known as collaborative tagging, social classification, social indexing and social tagging. The resulting assemblage of tags form a "folksonomy": a conflation of the worlds 'folk' and 'taxonomy' used to refer to an informal, organic assemblage of related terminology.

The systems that use folksonomies are mainly web based in which users can upload their resources and make labels on them, the so-called tags. In this way the system classifies and categorizes these resources based on user tags. Once a user is

logged in a system, he/she can view the resources of other users and optionally make tags to them. Resources can be anything even a system directed at a certain topic i.e. BibSonomy[2] resources are bookmarks, Flickr[3] are photos, last.fm[4] music files and Youtube[5] videos.

The collection of a user's assignments is called personomy and the collection of all personomies is called folksonomy. A user can explore personomies of other users and can tag other users' resources. Moreover when he/she clicks on a resource he/she can see which other users have tagged this resource e.t.c

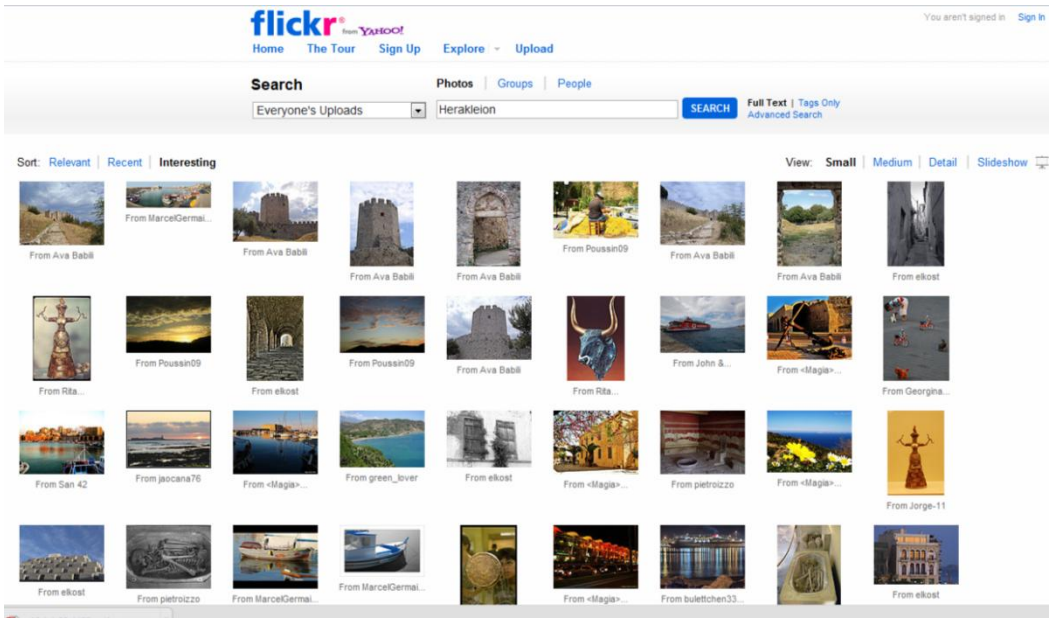


Figure 2. 9 Flickr Interface

Figure 2.8 shows the Flickr's search interface for Herakleion keyword. We can see a lot of images which have tags for Herakleion that users has created. In figure 2.9 we can see a photo and in the right part of the interface we see the tags for this photo which are Herakleion, Crete, Tonemaped , Geotagged.

[2] http://www.bibsonomy.org/
[3] http://www.flickr.com/
[4] http://www.last.fm/
[5] http://www.youtube.com/

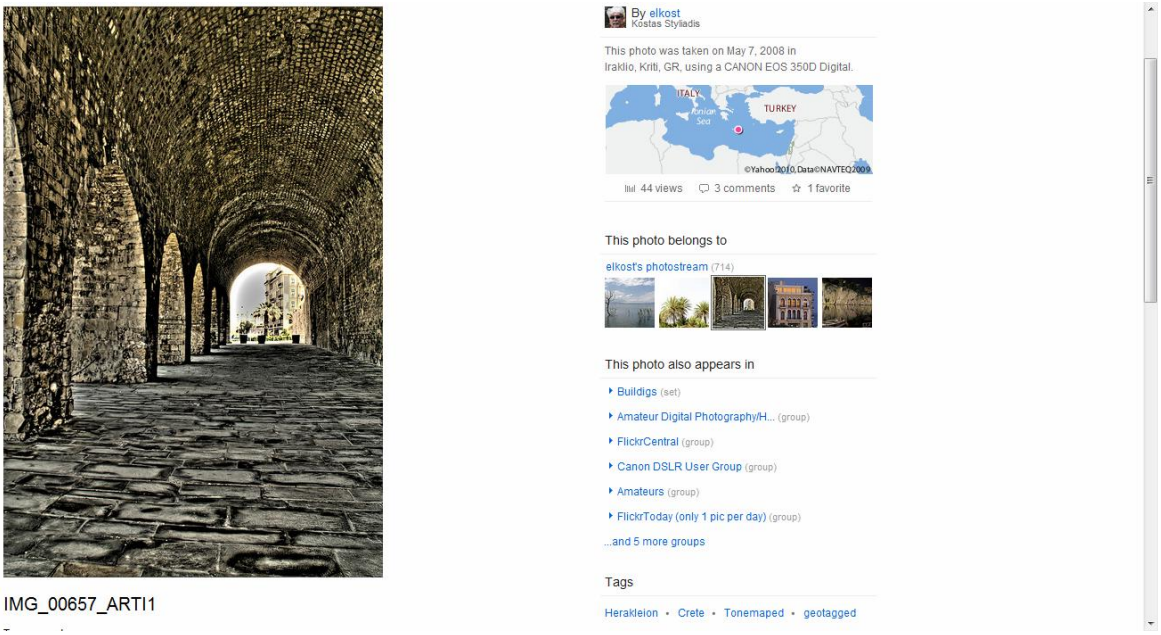


Figure 2. 10 Flickr interface

Many art museums aim at encouraging interaction between the users and at enabling them to make tags for their exhibits which are in digital form. The users can see the exhibits and if they have online access they may make tags. In Fig 2.9 we can see the web page of Cleveland Museum of Art[6] where we can find exhibits for tagging. In this case we make the Tag "Theotokopoulos" to the painting "Christ on the Cross"

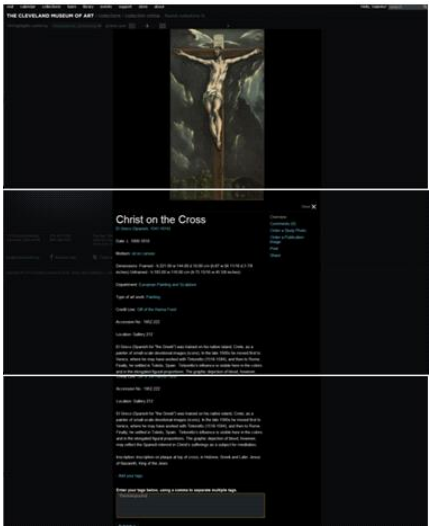


Figure 2. 11 Web site of Cleveland Museum of Art[7]

[6] http://www.clevelandart.org

[7] the photo is merge of three photos because it demands 3 screens in order to represent.

2.2 Summary and Discussion

In this section we have made comparisons of various systems and approaches for managing and preserving co-reference relations. In approaches for named Entity recognition that we described firstly, most of the works deal with text documents and the recognition of Entities in these documents. The success rates of NER are limited, it works well only in rich texts. In addition, there are cases that cannot define the type of an Entity that is referred in the text. Moreover, these approaches can easily work in a specific and strict domain but not necessarily with new domains and languages. In many cases the purpose of named Entity recognition is to convert an identification problem into a classification problem using heuristic and grammatical rules. Moreover it demands enough complete and comparable data which are difficult to find.

In the second part, we analyzed the co-reference systems such as GATE, LbjNerTagger, CRFClassifier which also deal mainly with linguistics. In these systems the main functionality is the recognition of specific Entities and the classification of them. Moreover, they do not provide solutions for other types of documents such as images. In this category the systems OOF (proposal of this system was available) and CRF were very interesting. The first system has limited range of applications because it deals only with biological terms. Furthermore, the CRF system creates bundles of URLs which are related to a certain Entity but the interface is very limited because it does not provide the documents or the areas of the documents in which these entities are cited. This system provides limited metadata and descriptions for a named entity and more specifically it uses only a name and the URL of the document. This method perhaps creates confusion because a user may associate the URL with an entity and not with the document or area of it. Finally CRF mainly deals with the classification of references rather than the identification of Entities.

The authority files as we described use methods of Named Entity recognition and as a result there are cases that they fail to recognize a specific Entity. We described examples in the previous section in which LEAF and VIAF try to identify person from birth and death dates e.t.c and as a result there are cases that these methods fail. Moreover, the authority files determine the co- reference between

documents for a specific named entity but do not provide a description for it in order to help in the identification and the determination of this entity.

The last category we described was folksonomies. In these applications, the users collaboratively tag resources on the web. These applications do not aim at making identity determinations of an entity but only deal with the classification and categorization of them.

To sum up, it has been made clear that the systems and approaches that we described, specifically the authority files, the NER and the NER systems which have automatic recognition algorithms, cannot work perfectly in all cases. They demand enough complete and comparable data which are difficult to find. Due to this reason, it is important to have a manual way to identify named entities. However, all existing automated methods are a completely different solution from the one we suggest in the next chapters which may be used to support or enhance the identity determination of entities. In solutions like NER systems and Folksonomies the aim is the classification of entities and not the identification. Moreover, NER systems do not provide a useful interface in which a user can monitor the description of Named entities and the citation of a Named entity in documents in order to help him/her make him/her own identity assumptions. In many cases, the only property which is available for the citation is the URL. The methodologies already outlined do not support a variety of documents like images, PDF files and HTML pages.

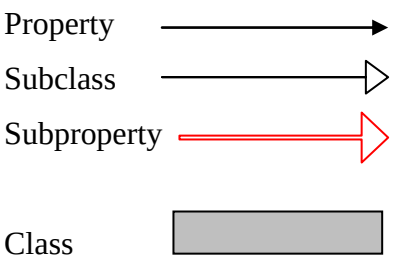
Chapter 3

3 Co-Reference and Identification Model

3.1 Introduction

In this Chapter we describe a model for Named Entities descriptions, Co-Reference and Identity Assumptions named CRIM (Co-Reference and Identification Model - see Appendix A.1). This model aims to represent the users' descriptions for named entities, specifically for Persons, Places and Events. Furthermore, CRIM models the citations in documents and the Identity assumptions of users for named entities which co-refer to different documents. The challenge for this model is the distinction between the different knowledge that users express about descriptions and identity assumptions for named entities. This model is extendable if we want to use other types of documents and other types of named entities.

In this chapter we will present parts of the model and we will show the most important aspects of it. In this model we "borrow" terms of CIDOC CRM (ISO21127)[41]. In the figures below the class names which start with the letter "E" i.e *E39 Actor* and the property names which start with the letter "P" i.e *P107 is current or former member of* correspond to CIDOC classes and properties, respectively. We present namespaces in pictures and we use the follow arrows to define the relation between classes:



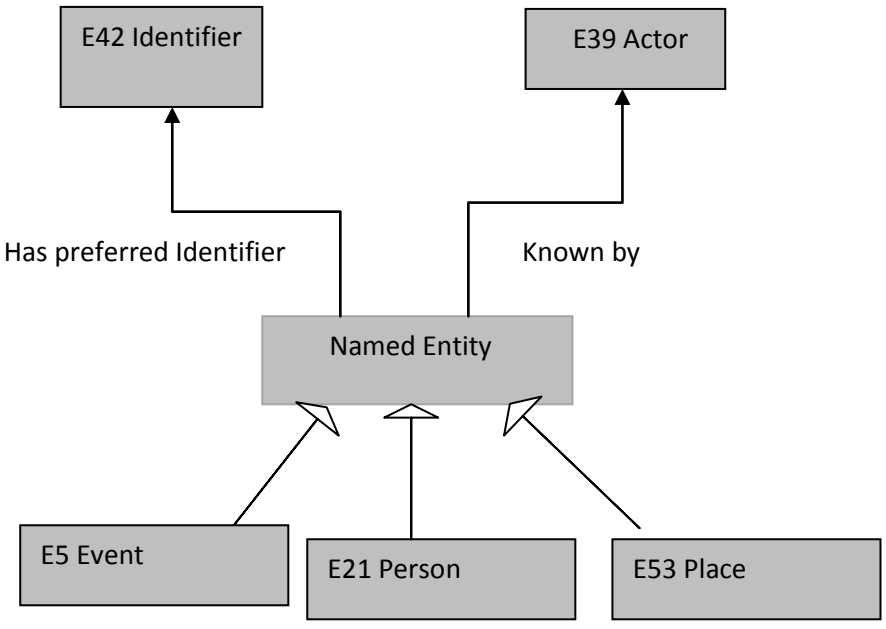
The rectangle of a class may have different colors.

The CRIM model has been developed on RDF for reasons which are described in section 3.8 and has been evaluated by the W3C RDF validation service[8].

3.2 Assumptions and basic terms

Apparently, we observe in the real world proper names are given to define entities such as persons, objects, places and events i.e. first and last names, place names e.t.c. *We define as named entity an entity in which we can give a proper name*. In this model we limit the term named entity only to persons, events and places. As we can see in figure 3.1 the classes Event, Person and Places are subclasses of a named entity and inherit its attributes.

We consider as *Actor, a person or group of persons who express his/her/their descriptions and knowledge for the named entities*. This model is limited only to Actor's knowledge which provides supporting information for the identification of a named entity. A named Entity must be known by an Actor who is responsible to define the attributes and elements for the entity. A named entity is known only by one Actor. The identifiers of a named entity are specified by an actor. A named entity may have multiple identifiers which identify the existence of the named entity in libraries and authority files.



[8] http://www.w3.org/RDF/Validator/ARPServlet

Figure 3. 1 Hierarchy of Named Entity

We give the definitions of a class of person, place, and event as defined in CIDOC CRM [41]:

The class *person* refers to a real person whose existence is known or it is assumed that they have lived.

The class *place* comprises extents in space, in particular on the surface of the earth. Furthermore it contains immobile objects such as buildings, cities, mountains e.t.c.

The *Event* class comprises of changes of states in cultural, social or physical systems, regardless of scale, brought about by a series or group of coherent physical, cultural, technological or legal phenomena.

3.3 Web Documents

It is important in our model to introduce terms such as Information objects in order to define the documents in which the named entities are cited. In our model the Information objects are documents on the web in which we can define areas (see 3.4 Areas) in order to describe the part of the document where a named entity is cited.

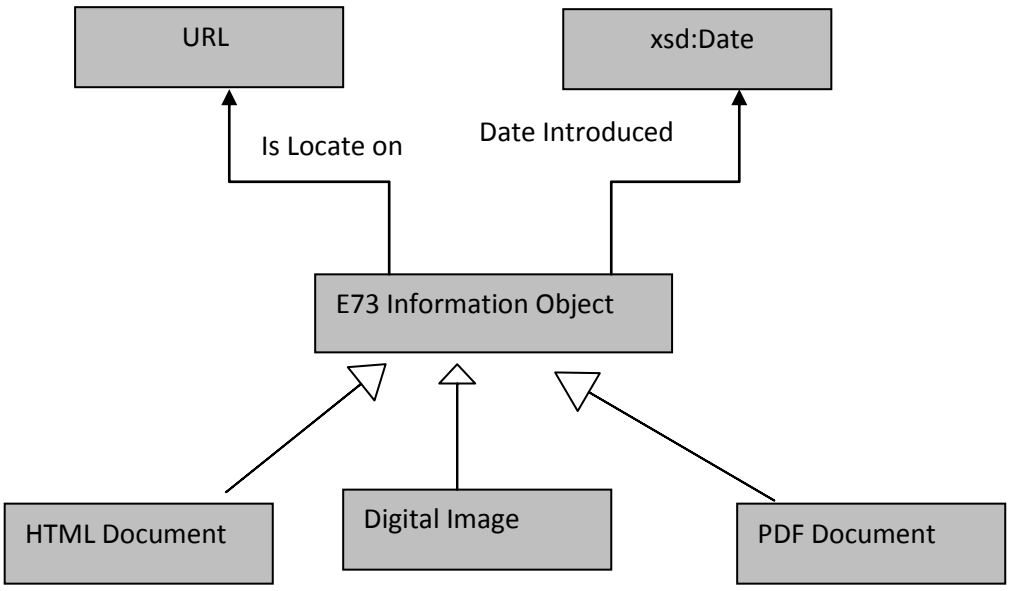


Figure 3. 2 Types of Infromation documents

The Information Object class is the domain of properties *is Locate on* and *Date introduced*. The property *is locate on* has as range the class URL (Uniform Resource locator) that specifies where an identified resource is available on the web. The domain of property *date Introduced* defines the date on which the document was introduced by the Actor. Furthermore the document may be modified in the future and this is a way of keeping a record of the state of the document in order to be able to search through it in web archives or in Memento[9].

The class Information object is specialized in 3 types (fig. 3.2):

- HTML Documents are pages on the web which mainly include text that cite the named entities.
- Digital Images are pictures available on the web which portray the named entities
- PDF Documents include text, diagrams or pictures which refer to named entities.

Finally we define as different the documents which have the same URL but have different access dates.

3.4 Areas on web documents

In this section we describe the areas on web documents. We use these areas in order to define the parts of documents that cite the named entities. Due to the diversity of documents, the areas on them are defined in different ways. In fig 3.3 we can see that the class Area is specialized in 3 classes:

- HTML area corresponds to HTML pages
- 2D area corresponds to 2D pictures
- PDF area corresponds to PDF documents

[9] http://www.mementoweb.org/

We note that in documents such as PDF and HTML which mainly include text that is possibly repeated multiply, it is not enough to only find the text but it is important to define the exact area where we can find the text in the document. In section 4 we provide a detailed analysis on how to define a certain areas in documents.

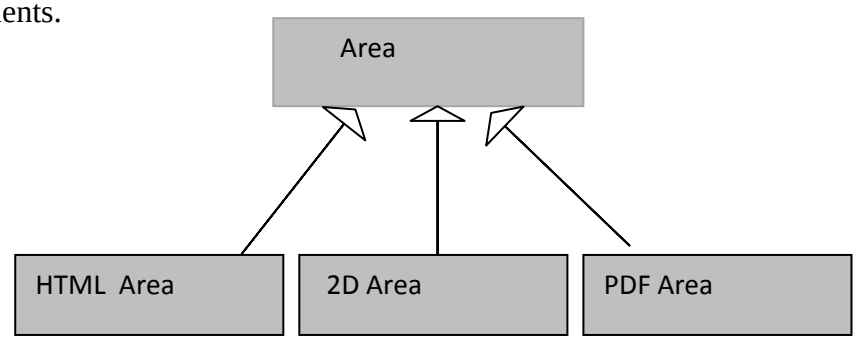


Figure 3. 3 Types of document Areas

In one document an actor defines one or more areas which cite a named entity. The property *Included in(fig, 3.4)* has as a domain the class Area and as range the class Information object. Furthermore it has 3 subproperties (*Included in PDF document*, *Included in digital Image* and *Included in HTML PAGE*) depending on the type of documents. These subproperties have as a domain the subclasses of Area class and as range the subclasses of Information Object.

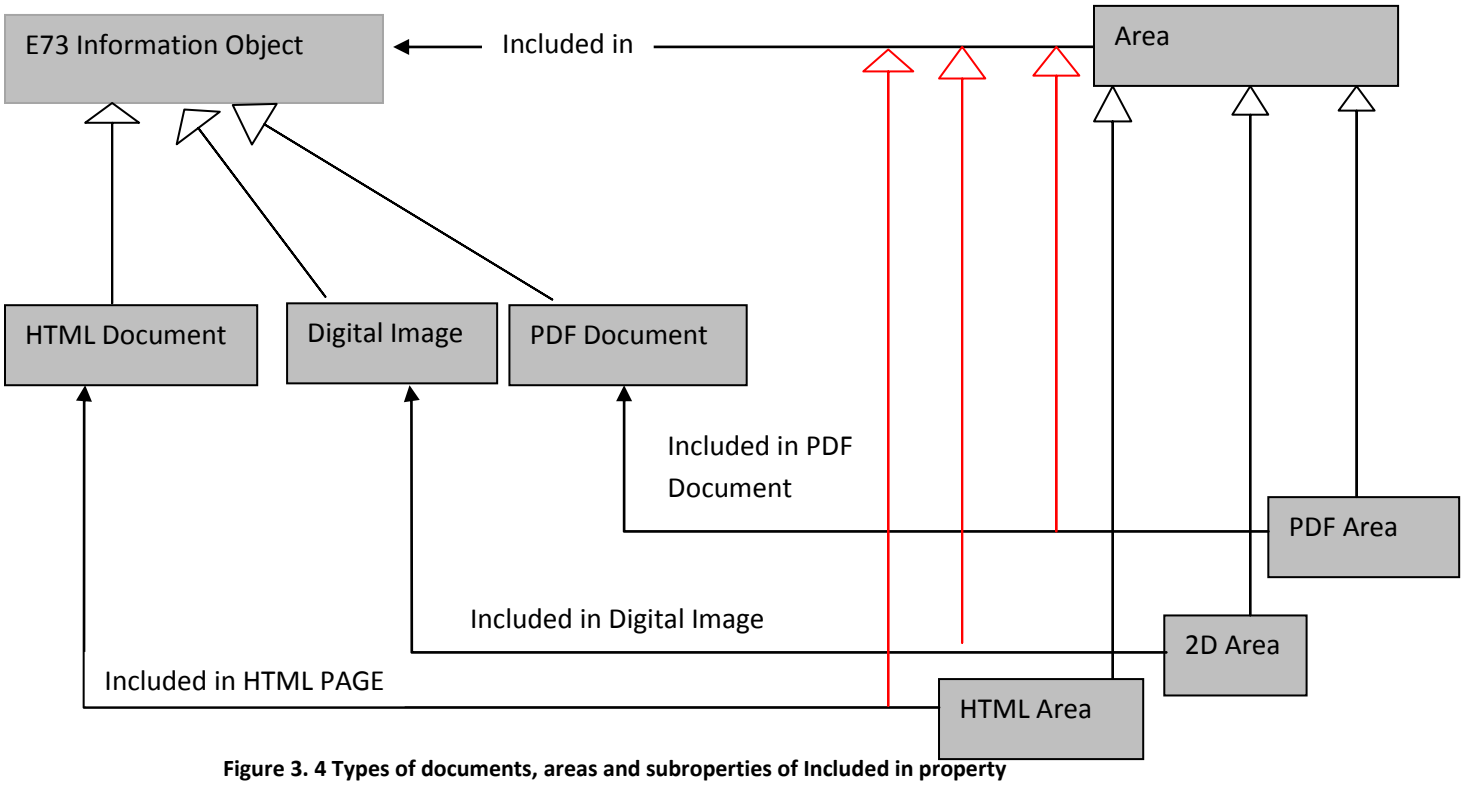


Figure 3. 4 Types of documents, areas and subroperties of Included in property

3.5 Citations

In the previous section we described the areas on documents that cite the named entities. In this model, as citation we define *the determination of an area on the Information Object that refers to a Named Entity*. If the information Object is an image, as citation we define the *determination of an area on the image that shows a Named Entity*. In fig 3.5 we present a citation that determines an area on a document and cites a Named entity. A citation determines only one area on a document and cites only one named Entity. It is based on users' knowledge which determines the specific area in a document where the named entity is cited. A citation is supported by an Actor which has determined it on a specific Date.

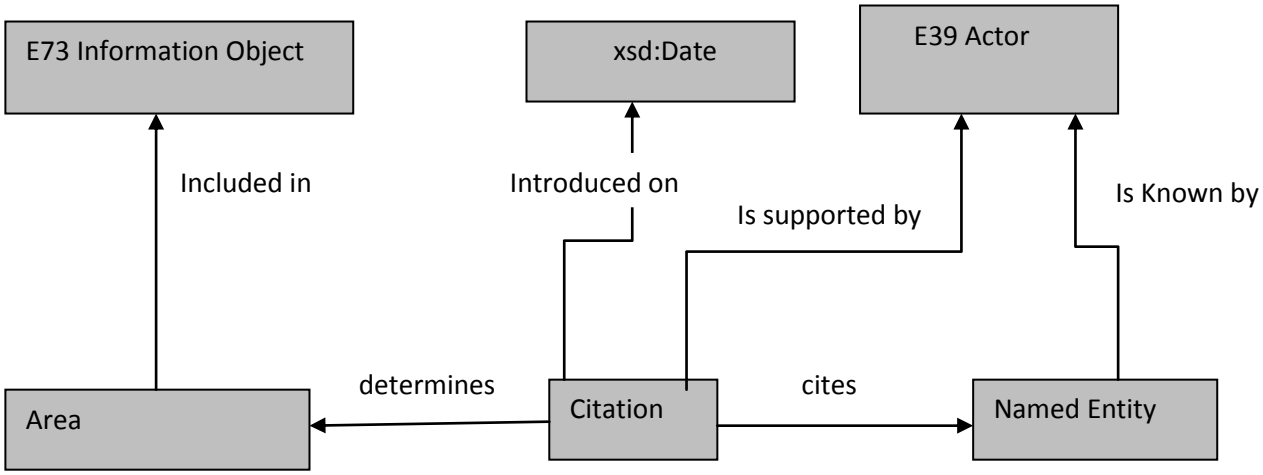


Figure 3. 5 The citation which determine a Document Area and cites a named Entity

As described in figure 3.4 the Area class is a super class of the PDF Area, 2Darea and PDF area classes and so a citation is independent from the area and the document type.

3.6 Co-reference

In linguistics, co-reference occurs when multiple expressions in a sentence or document refer to the same thing. In this model we define as *co-reference the reference of a named entity on multiple documents of the same or different type*. The

reference of named entities is determined not only by expressions or phrases but also by areas on the documents.

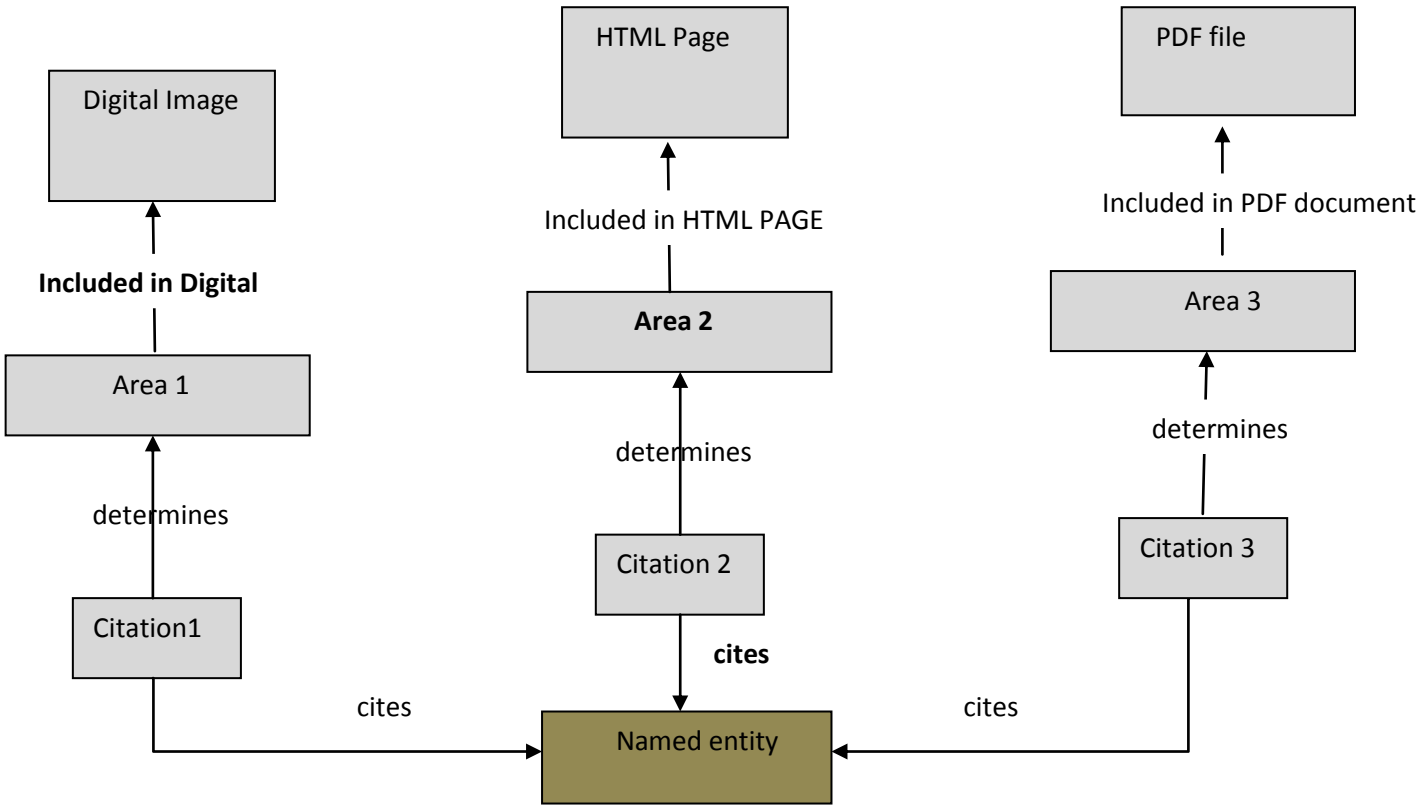


Figure 3. 6 Co-reference representation

In figure 3.6 a named entity is referred in 3 types of documents. The first citation determines a citation on a digital image, the second on an HTML page and third on a PDF file.

3.7 Identity Assumptions

In the previous sections we showed that the descriptions of named entities and the citations of them on documents, came from the knowledge of users. It is auxiliary to collect the users' knowledge because the descriptions of the named entities operate supportively in order to help other actors determine the identity of named entities. In this model we aim at representing the assumptions of actors about the identity of the

Named Entities. Specifically, the actors infer the possible identity from the similarity or the difference of the properties which are available from the descriptions of the named entities. The Actors know the identity by the context of citations or the content of the entity's description. The process that *an Actor determines an assumption about the identity of two named entities, we define it as Identity Assumption*. One Actor can create none, one or many assumptions but only one for two specific entities. As described in [42] it is very difficult to define the identity. In addition, the ubiquitous owl:sameAs property is used as the RDF property to connect data-sets and it has been dubbed the owl:sameAs problem.

The obvious types of identification are the similarity and difference of two named entities but we also use the possible similarity type if an actor is not sure about the identity of two named entities. As described in figure 3.6 an actor "creates" an Identity Assumption for two named entities which identifies them as same, not same or possible same.

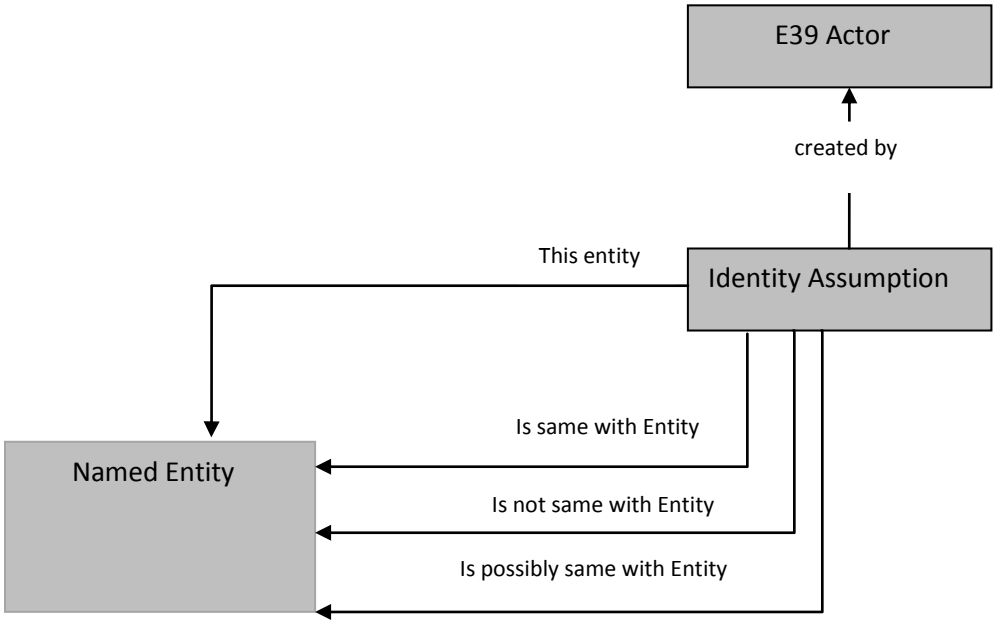


Figure 3. 7 Types of identification

In section 3.2 we described that a named entity has 3 types: persons, places and events. Also in this thesis the identity assumptions are limited only to persons and places because the citation of the events does not help in historical analysis

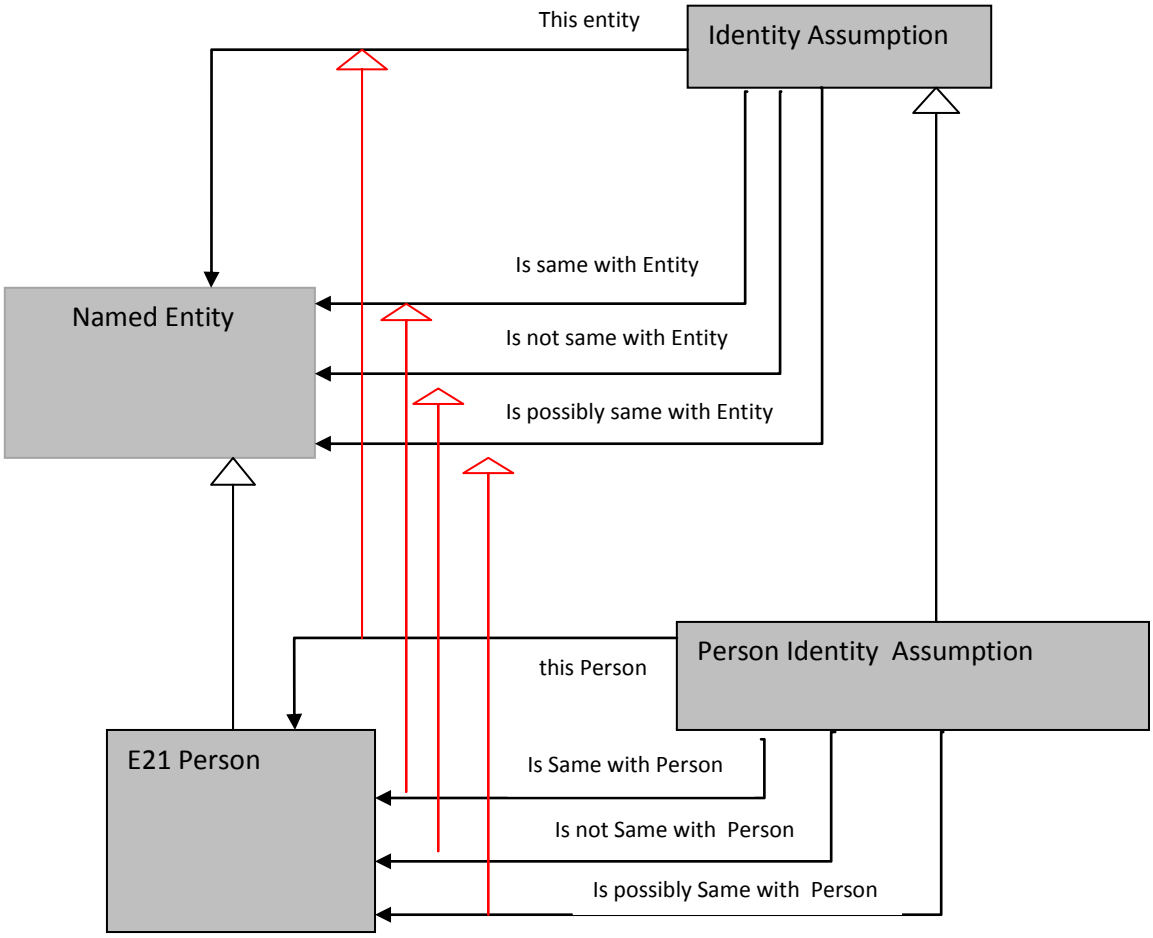


Figure 3. 8 Person Identification

In figure 3.8 we can see the types of Identity assumptions for persons. The class Person Identity assumption is a subclass of Identity Assumption and also has 3 types of identification (same, not same and possibly same) which are sub-properties of the respective properties of Identity assumption.

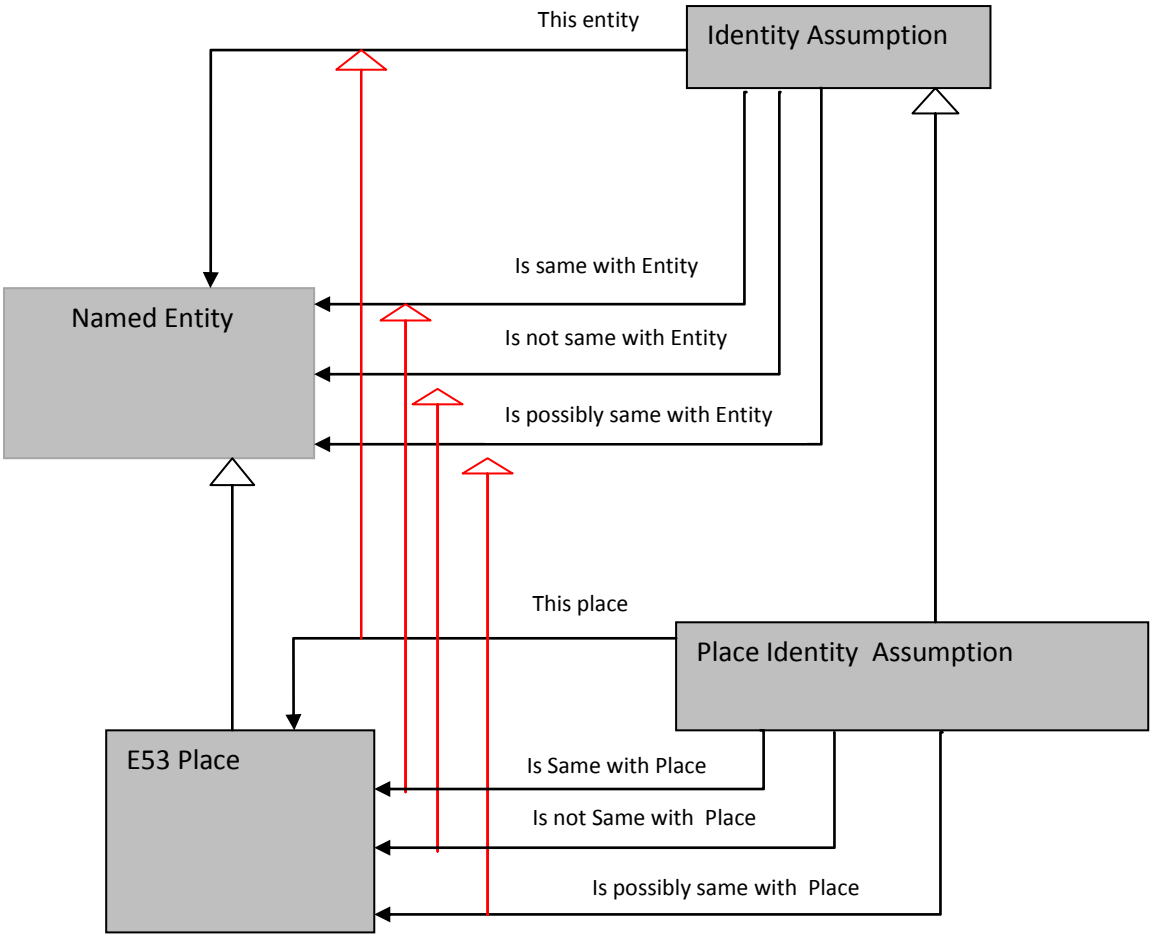


Figure 3. 9 Place Identification

In figure 3.9 we can see the types of identity assumptions for places. The class Place Identity assumption is a subclass of Identity Assumption and has also 3 types of identification (same, not same and possibly same) which are sub-properties of the respective properties of Identity assumption.

In figure 3.10 we can see an example of Identity Assumptions. In this example the Actor John knows the place which is called Herakleion and the Actor Nick knows a place which is called Handax. The user John makes an assumption in which he believes that the place which is called Herakleion is possibly same with the place which is called Handax. In addition the Actor Nick makes an assumption in which he believes that the place which is called Handax is the same with the place which is called Herakleion.

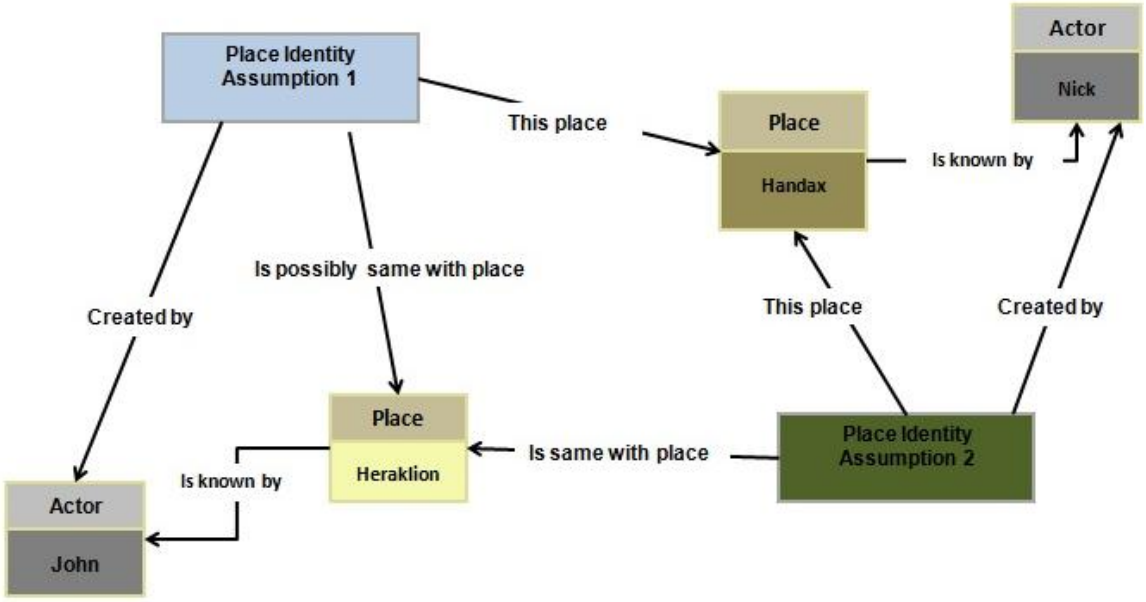


Figure 3. 10 Example of Identity assumption

3.8 Benefits using RDF vs. XML

In this chapter we analyze the benefits of RDF[10] in comparison to XML[11]. The implementation of CRIM is based on RDF and RDFS for the following reasons:

1. RDF is based on the creation of interconnected statements which easily represent identity assumptions and co-reference than XML which is based on the creation of documents.

2. RDF is graph structured whereas XML is tree-structured. The networks which are created in identity assumption and co-reference follow the graph concept and do not follow a specific order which is in tree based concepts.

3. Updating an XML document requires the maintenance of the tree structure and the node's order. The order of RDF properties is not specific while it is in XML.

4. The triple model of RDF simplifies the way that semantics are represented and described whereas for XML there will be a lot of documents which represent semantically the same structure.

[10] http://www.w3schools.com/rdf/rdf_intro.asp
[11] http://www.w3.org/XML/

5. In RDF we can easily present external relationships between entities. On the other hand, the XML document, only presents relationships between different elements on the specific document applying strictly the hierarchy in the tree. There are no external relationships that can tie two entities directly.

6. The metadata for the named Entities, the identity Assumptions and the co-reference representation change dynamically, independently of a certain document or context.

7. RDF expresses semantic meaning and knowledge contrary to XML which expresses syntactic meaning alone.

Chapter 4

4 WhoWhereWhen: A platform for the management and collaborative identification of co-reference on digital resources

4.1 Introduction

In this chapter we describe and analyze our system and its architecture. The platform we have developed, is named WhoWhereWhen (WWW) and is a web based information system which uses ideas of WEB 2.0. It uses the technologies of the semantic web in order to represent the user's descriptions and identity assumptions. The main functionalities of the WWW are the representation of co-reference between web documents, the introduction of descriptions by the users for the named entities and most importantly the management of identity assumptions of users for different named entities. The system is built on the CRIM model which is described in Chapter 3 and enables the distinction of the different knowledge that users express about descriptions and identity assumptions for named entities. The platform enables the user to monitor the co-reference between documents, to study the descriptions of other users that have been expressed for the named entities and then to make their own identity assumptions for different named entities. The challenge in this platform is the development of a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of document citations and associated identity assumptions.

The descriptions of the named entities but also identity assumptions of users are based on users' knowledge exclusively. The WhoWhereWhen platform aims to guide the users making their own identity assumptions, but does not aim to make interpretations for the named entities. The WWW does not confirm if the knowledge

which is introduced in the knowledge base is true. It does not aim to replace the users because the WWW is not a decision making system. It aims only at representing the knowledge which has been entered in the system through an intuitive interface that guides the user in expressing his/her assumptions about the named entities and identifications of them.

In the last section of this chapter we present an example which is based on real conditions 4.4.4.2.

The Interface of the system aims to guide the user in distinguishing the notions and the associative citations that other users have introduced. The graphical representation of the citation on documents and the ergonomic representation of the named entities help the users easily study and understand the descriptions and the associated references on the different types of documents. The design of the interface aims to provide enough information in an intuitive way in order for the user to make his/her own identity assumptions for the named entities. Finally, it aims to provide the user with an easy navigation system in order to help him/her understand the co-reference between documents without using incomplete information or information which may disorientate or make it difficult for the user to complete his/her historical analysis.

Section 4.2 describes the assumptions and conditions under which the WWW works and how the latter is restricted by the model. In 4.3 we describe basic aspects and technical choices of the WWW architecture. In 4.4 we further describe the architecture, the layers of the system and important technical decisions we got. Moreover, in this section we describe the data flow in order to make thoroughly clear the architecture of WWW and the management of documents especially when it comes to storing and highlighting the areas on it where named entities have been found. Finally in 4.5 we present an example in order to describe the user interface in detail.

4.2 Assumptions and conditions

In this section we describe the assumptions and conditions under which the WWW system works. The system uses the CRIM (Chapter 3) which has been

developed as a theory and not as a functional database schema. The model aims at representing the knowledge that the users have for the named entities and the identity assumptions they make. However, it does not aim at solving conflicts if, for example, some users make different identity assumptions for a specific named entity. Therefore, the implementation of WWW which uses the CRIM follows the following specific assumptions and conditions:

1. In WWW the descriptions of named entities, the citations and the identity assumptions are based on actors' knowledge exclusively. The system does not determine if this knowledge is true or if this knowledge is objectively acceptable from other actors. If an actor disagrees with the theoretical interpretation of another actor, he/she may express his/her own knowledge about a named entity and then he/she can differentiate this named entity with the named entity that the other Actor has defined.

2. The identity types are restricted on three types: Same, Not Same, Possibly Same

3. The users describe and determine the identity only for persons and places.'

4. The platform determines the area on a document that cites the named entity and does not describe the content of it

5. The platform uses the Actors' descriptions for named entities in order to guide the actors in making their assumptions for entities. It does not make strict descriptions for them.

6. The WWW system represents the identity assumptions that Actors define but does not aim to resolve conflicts if two or more Actors make different types of identity assumptions for two specific named entities.

7. We assume that documents which refer to a named entity do not change or get modified. If a document changes or gets modified over the time we consider it as a different document. The main reason that we make this assumption is that

it is impossible to define again an area which is associated with a named entity if a document has been changed or modified.

8. The users know every fact of the system and are responsible for its monitoring if new named entities or identity assumptions have entered the system.

4.3 System description

In this section we describe and analyze the architecture of the WWW system. As we can see in figure 4.1 the WWW platform is stratified and it consists of 4 layers. Due to its stratification the platform can be easily modified and extended. Moreover in this chapter we explain the choices and the decisions we have taken in the development of the platform.

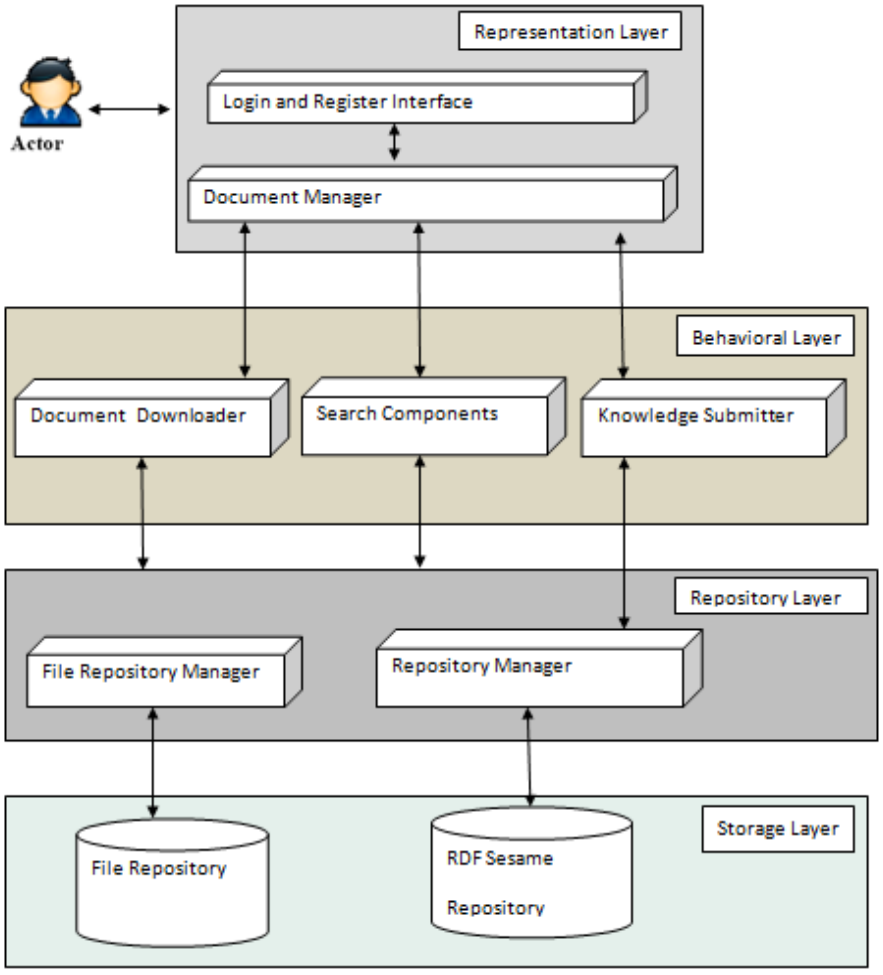


Figure 4. 1 WhoWhereWhen Platform Architecture

4.3.1 System Architecture

The architecture of the system is shown in figure 4.1 and is structured in 4 layers. Specifically the layers are:

1. **The representation layer**: This layer includes the user interface which is based on WEB technologies. The user interface is described more in 4.4.4.3 and contains dynamic web pages enabling the user to interact easily. In this layer the challenge was to develop a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of document citations and associated identity assumptions. Our interface has received influences from the ideas of WEB 2.0. In order to develop the pages we used JavaScript, JSP, Ajax, html and CSS. The main component of the interface's page is the Document Manager which has 3 types; the PDF manager, the Image Manager and the Page Manager which offer a specialized interface for each document.

2. **The Behavioral Layer:** This Layer is the core of WWW system because it contains the most important components and implements the logic of the system. It is responsible for the communication between the representation and the repository layer and it has been developed in JSP exclusively. It includes the Document downloader which consists of a PDF downloader, an Image Downloader and a Page downloader. There was the need to specialize the Document Downloader in 3 types because each type of document has different demands in its management. Also in these components the system runs queries in order to monitor the existing citations in the specific documents managed. The Search Components include all the JSP pages which enable a thorough search functionality such as Entity search, Citation Search and Assumption Search. Finally, the Knowledge submitter includes the JSP pages which enable users submit their knowledge about Named entities, Citations and Identity Assumptions.

3. **Repository Layer:** The WWW system includes 2 types of repositories. The first repository is the RDF sesame repository which keeps the knowledge of users and the File repository which is used in order to keep the documents in which the

users make citations. In this layer we have two components, specifically the Repository Manager which is related to the management of the RDF Repository and the File Repository Manager which is related to the File repository. The functionality of the two components is very different due to the different type of data they manage. The first component includes the functionalities of submitting RDF files assimilated with users' knowledge and making queries. The second component includes the functionalities of downloading the document to the repository, keeping archived files and normalizing their URLs (4.4.2.4). The components of this layer have been exclusively developed in java.

4. **Storage Layer:** This layer includes the RDF sesame repository which keeps the knowledge of users and the file repository where the documents are being downloaded.

4.4 System Platform and components

In this section we further analyze the components that the WhoWhereWhen (WWW) platform consists of. As we have described in the previous section the platform consists of four layers: Representation Layer, Behavioral Layer, Repository Layer and Storage Layer. We will start the analysis from the lowest layer which is the Storage Layer.

4.4.1 Storage Layer

This Layer consists of two repositories which keep different types of data. Specifically the repositories are the RDF Sesame repository which keeps the knowledge of the users and the second is the File Repository which keeps the documents the WhoWhereWhen system manages in order to represent the citations. Below we analyze the repositories and the reasons for our choices.

4.4.1.1 RDF Sesame Repository

The WWW platform uses a Sesame[12] RDF repository in order to retrieve and store RDF statements. It is an open source Java framework which makes it possible not only to store RDF statements but also to query and reason with the RDF and the RDF schema. The installation of Sesame is very simple and is incorporated in Apache Tomcat[13] which is an open source software implementation of the Java Servlet and JavaServer Pages technologies. The Sesame is used as the Server that receives requests from clients via the http protocol. It uses web services for clients to store, retrieve and query RDF statements easily.

The Sesame provides useful functionality. Firstly, it provides a variety of different repositories in order to store RDF Data. Specifically it can store data in main – memory, on disk and in a relational database. The first type of repository is the fastest because it uses the main memory contrary to the others which use the disc in order to store and query RDF data. The Native RDF Repository which keeps and retrieves data works better on disc in large datasets since the memory is enough to store it and its size is not limited something that does not necessarily stand for the main-memory. In the WWW we cannot define the size of RDF statements which will be stored by the users and for this reason the Main memory repository was not useful. Moreover, it was not safe to keep the data in memory because if we shut down the server the data could have been lost. So, for the reasons we described above it was useful to use the Native RDF repository which stores the data directly on the disk.

Also Sesame provides a very useful and powerful API (Application Programming interface) in Java Language which is compatible with the Tomcat server and with the WWW platform. It provides a lot of applications for managing and manipulating Sesame repositories as well as functionalities like RDF parsers writers and storing functionalities. Moreover, it provides a lot of methods in order to create, delete and submit statements easily.

In addition Sesame provides a lot of mechanisms which enable querying in a lot of RDF query languages specifically in SPRQL and SERQL and offer various

[12] http://www.openrdf.org/
[13] http://tomcat.apache.org/

output formats for query results (XML, HTML, RDF statements). Moreover, Sesame supports named graphs which help us define the context of specific statements. The context is very helpful in order to insert, delete and edit the set of statements which are identified from a certain context identifier.

Finally, the Sesame provides the Workbench which is a graphical tool in order to manage the Sesame server. It is a very useful functionality because it enables developers monitor the state of the RDF repository. Moreover, it provides a lot of applications which are very useful in testing the repository at any time throughout its development.

4.4.1.2 Query Language

The WWW platform as we have described, represents the users' knowledge in RDF and it needs an RDF query language in order to retrieve these data. The query language we use is SPARQL[14] which is a W3C standard and is supported by the Sesame. SPARQL is an expressive language and it has those capabilities for querying that serve our purposes along with optional graph patterns including their conjunctions and disjunctions. SPARQL also supports extensible value testing and constraining queries by a source RDF graph. The results of SPARQL queries can be result sets or RDF graphs. It follows a brief description.

At most SPARQL queries contain triple patterns which represent basic RDF graphs. Each element of the triple pattern (subject, predicate and object) may be a variable. Each variable before its name has the "?" character. A simple SPARQL query it consists of the select clause which defines the variables of which we want to get the instances and determines the order that we want to get the results. The WHERE clause provides the basic graph pattern to match against the data graph. The following example retrieves the URL of the specific document that has been stored in the RDF repository. The variable that exists in this query has the name ?url.

[14] http://www.w3.org/TR/rdf-sparql-query/

PREFIX xsd:<http://www.w3.org/2001/XMLSchema#>

PREFIX rdf:<http://www.w3.org/1999/02/22-rdf-syntax-ns#>

PREFIX coref:<file://corefV4.rdfs#>

SELECT DISTINCT ?url

WHERE { coref:Document coref:CR_Is_Locate_on ?url}

A lot of query languages for RDF are available. The most common languages are SPARQL, SERQL and RQL. The SeRQL (Sesame RDF Query Language) is available on sesame but the SPARQL is available on a lot of RDF repositories such as Jena, Redland, 3Store, Sesame and SPRQL is a W3C standard. These were the main reasons that we choose the SPARQL because it is helpful in case we want to change the sesame in order to use another RDF repository in the future.

4.4.1.3 File Repository

The second repository which the WhoWhereWhen system uses is very different from the first due to the different type of data that keeps. This repository is relative to documents that are downloaded for caching (see 4.4.2.3). This repository stores the html files, the PDF files and Images that we use to represent the citations of named entities. We choose these types of documents because they are the most popular in historical research. Moreover we store the prime page of PDF files having, first, converted it to a PNG image. We perform the same for the other pages of PDF files only at execution time (see 4.4.4.1.2). The URLs of the documents are normalized for the reasons we describe in 4.4.2.4 before storing the documents.

4.4.2 Repository Layer

This Layer is in the middle between the Storage and the Behavioral Layer and it consists of two components which manage the repositories that we described above. The first component is the Repository Manager which contains methods that manage the RDF sesame repository. The second component is the File Repository Manager which includes java classes that manage the File Repository in which web documents

are stored and below we further analyze the functionalities and the decisions we have made regarding the use of the file repository. The components are very different and provide different functionalities due to the different types of data they manage. This layer has been exclusively developed in JAVA.

4.4.2.1 Repository Manager

This component is one of the most important parts of the WhoWhereWhen platform. It provides the methods in order to store, retrieve and delete RDF statements from the Sesame RDF repository.

It provides methods which enable the connection with remote Sesame repositories. As we described in 4.4.1.1 the Sesame is a server which is installed in apache tomcat. This server receives and sends requests in web services via the HTTP protocol in order to store new knowledge in RDF or answer to queries which retrieve the storing knowledge.

In order to query the repository the following methods are provided:

- **runSPARQL2GRAPHString**:Executes a Construct/describe SPARQL query against the graph. It returns a serialized graph as String.
- **runSPARQL2GRAPHList**: :Executes a Construct/describe SPARQL query against the graph. It returns a serialized graph as List.
- **runSPARQL2List**:Executes a SELECT SPARQL query against the graph. It returns a list of solutions, each containing a hashmap of bindings.
- **runSPARQL2XMLString**: Executes a SELECT SPARQL query against the graph and returns a SPARQLXML result of solutions, in String object.

In order to remove context and statements the following methods are provided:

- **clearContext**: Clears the context from its data
- **remove**: Removes the supplied statement from the specified contexts in the repository.
- **clearAllContexts**: Clears all contexts from the repository.

In order to introduce new knowledge in RDF format the following methods are provided:

- **addTriple:** Inserts a triple into a graph
- **addTripleStatement**: Inserts a triple/statement into a graph
- **addString**: Imports RDF data from a String
- **addStream**: Imports RDF data from a stream
- **addFile**: Imports RDF data from a file
- **addURI**: Imports data from URI source

4.4.2.2 File Repository Manager

This component provides the functionality of managing the file repository. As we described in 4.4.1.3 the file repository stores the documents that Actors have introduced and downloaded along with their URL. Regarding downloading, 3 types of documents are supported specifically Images, HTML pages and PDF files because they are the most popular in historical research. The repository includes four folders which are:

- **PDF** folder which stores PDF files
- **HTML** folder which saves the HTML pages
- **Images** which saves the image files
- **Page Pictures** which is a subfolder within the PDF folder and saves the PDF pages as images in PNG format (see 4.4.4.1.2).

The main reason that the File Repository is created is the caching of documents (see 4.4.2.3). The file name of documents that are stored in the file repository comes from their URL which is normalized for reasons that we describe in 4.4.2.4. Due to the different type of documents that are stored in the repository, it is necessary to download them in a different way (see 4.4.2.5). Finally, as we have outlined in section 3.3 two web documents with the same URL but different dates of access are considered different.

4.4.2.3 Caching

This process is known as web caching and is used for caching Web documents. This process includes the downloading of web files (specifically pages, images and PDF files) and their storing to the File repository. As we have outlined in section 3.3, two web documents with the same URL but different dates of access are considered different. The main reason that we use this assumption is that if a user has defined an area on the document which is associated with a specific named entity, it will be difficult to define it again if the document gets changed or modified. Moreover, the definition of areas in PDF and html files is only possible when the documents have been downloaded. Summarizing the main reasons that lead us to use this process are:

1. The reduction of bandwidth in order to open a document, specifically in large PDF files and web pages
2. The delay if we download a document more than once.
3. The same origin policy(see 4.4.4.1.3) and the management of PDF files(see 4.4.4.1.3)
4. The demand of huge amount of bandwidth in entity search (see 4.4.3.2.1) when the results containing a large number of documents.

In addition the usage of caching is important when we use PDF documents. As we describe in 4.4.4.1.2 we convert the pages from PDF files to PNG images. In this case the caching is necessary because a user may read a lot of pages and it will be useful to create only once the image, regardless of how many times the user reads this page. The pages that a user has read, is deleted when the user closes the page of the PDF manager (described in 4.4.4.1.2)

In the future, if a lot of users use, introduce and download web documents in the WhoWhereWhen platform the repository will store a large number of documents and this will cause problems if the repository requires a very large hard disk space. In this case it will be useful to have a different approach to the management of the file repository.

One solution is the usage of web archives[15] which store old versions of web documents. A recent solution is the Memento Project[16]. If you know the URI of a Web resource, the technical framework proposed by Memento allows you to see a version of that resource as it existed at some date in the past, by entering that URI in your browser like you always do and by specifying the desired date in a browser plug-in. Obviously, this will only work if previous versions are available somewhere on the Web. But if they are, and if they are on servers that support the Memento framework, you will get to them. In the future, the utilization of Memento may become an integral part of our work but for the time being, as it is a new approach, we will let it mature. Also, some compatibility problems with the WhoWhereWhen do not allow the usage of them immediately.

4.4.2.4 Normalization

The normalization[17] of URLs (or URL canonicalization) is a very important process in the caching of web documents. It is a common and necessary process in web crawlers[18] used by search engines. The goal of the normalization process is to transform a URL into a normalized or canonical URL so it is possible to determine if two syntactically different URLs can be equivalent. The WhoWhereWhen platform employs URL normalization in order to:

- Reduce downloading of duplicate documents.
- Avoid downloading the same document more than once.
- Determine if a document has already been cached.
- Modify characters on the URLs which are not acceptable from the file system of the operating system that stores the repository.

[15] http://en.wikipedia.org/wiki/Web_archiving
[16] http://www.mementoweb.org/
[17] http://en.wikipedia.org/wiki/URL_normalization
[18] http://en.wikipedia.org/wiki/Web_crawler

In the WhoWhereWhen platform the normalization process makes these changes to the URL names:

1. **Converting the scheme and host to lower case.** Example:
   HTTP://www.Example.com/ → http://www.example.com/

2. **Adding trailing**. Example: http://www.example.com →
   http://www.example.com/

3. **Removing directory index.** Example:
   http://www.example.com/default.asp → http://www.example.com/
   http://www.example.com/default.asp → http://www.example.com/

4. **Removing the fragment.** Example:
   http://www.example.com/bar.html#section1→
   http://www.example.com/bar.html

5. **Removing the default port.** Example:
   http://www.example.com:80/bar.htm → http://www.example.com/bar.html

6. **Removing dot-segments.** Example:
   http://www.example.com/../a/b/../c/./d.html→
   http://www.example.com/a/c/d.html
7. **Capitalizing letters in escape sequences.** Example:
   http://www.example.com/a%c2%b1 → http://www.example.com/a%C2% B1

These are the main changes that the WhoWhereWhen makes on URLs. For more info see http://en.wikipedia.org/wiki/URL_normalization.

During the development of the platform some problems appeared with the characters that the Windows operating system's file system does not allow. Specifically, the characters "/", ":", "?", "*", """, "<", ">", "|" are not allowed in file names and were changed with the character "_" . Moreover the space character " "

was changed with encoding "%20" because problems appeared with respect to the opening of documents via the http protocol.

4.4.2.5 Document downloading

The caching of web documents has as a prerequisite the downloading of them. The different types of documents require a different approach regarding their downloading. The algorithm that was used in order to download a web document is:

Algorithm 1

```
DownloadDocument(URL)
    n = new Normalization();
    FileName = n.normalization(URL);
    f = new File(FileName);
    If (f.exists() == false)
        Download();
    EndIf
```

Also the function Download() is different for each document due to the difference of type.

- **Images**

  The picture types which are compatible with the WWW platform are the JPG, and PNG format. We use the javax.swing.ImageIcon in order to get information which is relative with the image. The basic information that we need for the images is the width and height of them in order to show it on the interface without deforming them.

- **PDF**

The PDF documents are very common in historical research, for this reason it was appropriate to embed this functionality in the WWW platform. The PDF is a closed protocol and demands specialized classes in order to manage it. It is necessary to get information for a PDF file specifically the title of the document, the author and the number of pages. The platform uses the ICEpdf[19] an open source Java PDF engine that can render, convert, and/or extract PDF content within any Java application or on a Web server. Moreover we use the ICEpdf not only for extracting information for PDF files but also for other functionalities which are described in 4.4.4.1.2

- **HTML**

The determination of this type of files is not so clear because an html page may have been developed in multiple ways and it may also include a lot of other files and languages such as images, links and css files which control the style and layout of the pages. It is common to use external files in order to write the commands for the css stylesheet.

In addition, a web page may have contents which change dynamically at runtime. In the WhoWhereWhen platform we deal with the pages which have static content. The main challenge of this type is to download the web page keeping it at the same time immutable. In order to achieve this goal we take the following actions:

1. Check the paths that lead to css files and modify them where necessary.
2. Check the paths that lead to images and modify them where necessary.
3. Save the html pages in UTF-8 encoding in order to keep characters specifically in Greek or other languages immutable.

Finally, the change in appearance of an html file depends on the source code because there are cases in which the developer of the page leaves unmatched html tags and specifies the paths of any css files in an undefined

[19] http://www.icepdf.org/

way. In these cases it is very difficult to find the faults or to correct the paths in order to keep the page immutable.

4.4.3 Behavioral Layer

This layer is the core of the WhoWhereWhen platform because it consists of the most important components of the system and implements the logic and the behavior of the system. This layer is between the representation layer and the repository layer and is responsible for the communication between these layers. The main components of this layer are the document downloader which is relevant to PDF, Image and Page Downloader due to the different types of documents that a user may download. Also these components run queries relative to the documents and return the data to the representation layer in order to view the citations in the documents. The Search Components include Entity, Citation and Assumption Search. Finally, the Knowledge Submitter includes the components which enable the users to submit their knowledge about named Entities, Citations and Identity Assumptions. The components of this layer have been developed exclusively in JSP.

4.4.3.1 Document Downloader

As we described above this component includes JSP pages which use the methods of the repository layer in order to download a web document, to extract attributes of the document and finally run queries in order to define the citations in the document. Each Downloader is called from the Document Manager (described in 4.3.2) via Ajax calls and returns the data which are necessary in order to present it on the interface. Below we describe further each type of Document downloader.

The Document Downloader in each type receives the URL of the document. It checks if this document has been already downloaded on the file repository and extracts metadata of this. Then the Document Downloader makes queries for the named entities in order to get the descriptions and citations of them on the document. In figure 4.2 we can see the flow chart of the Document Downloader.

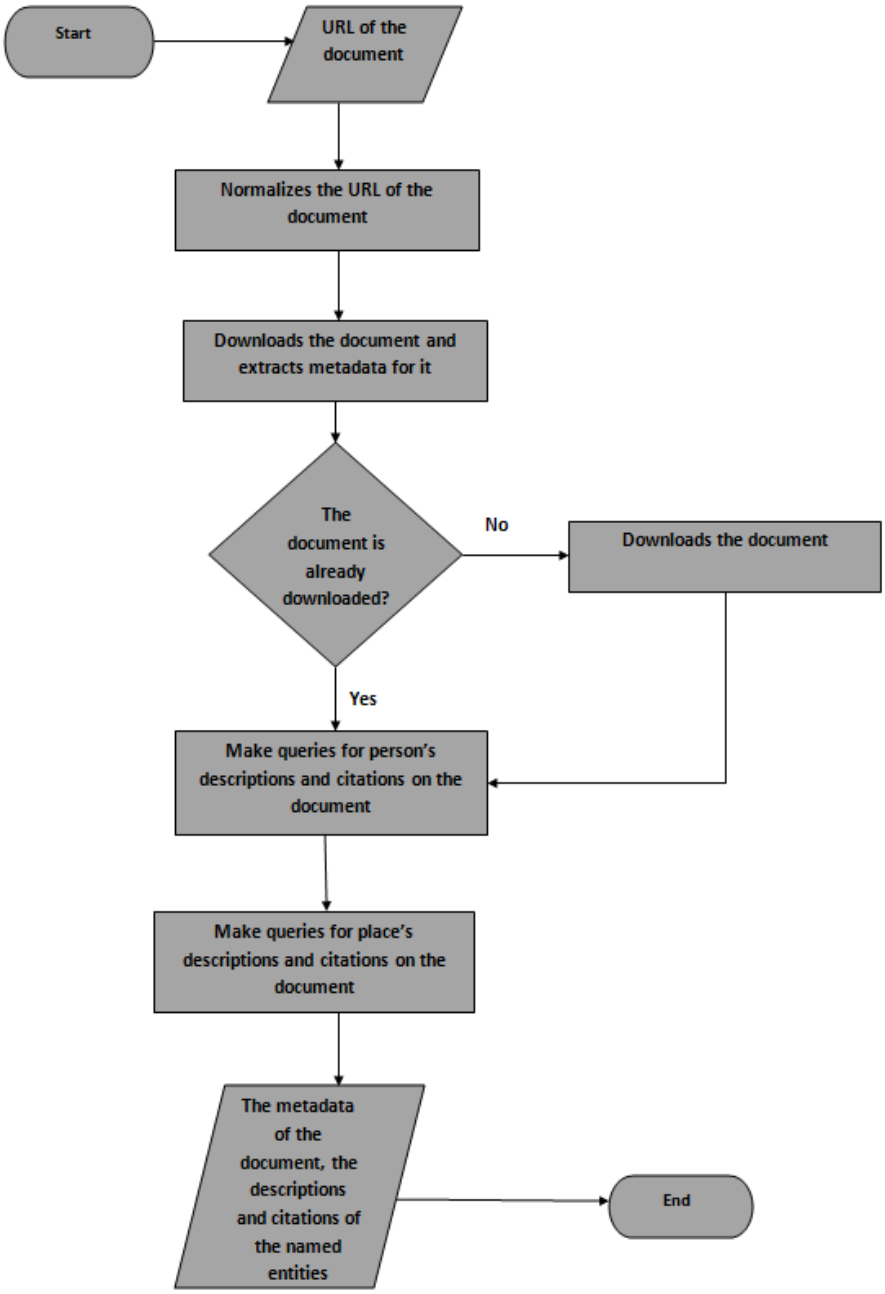


Figure 4. 2 The flowchart of Document downloader

4.4.3.1.1 Image Downloader

This downloader is called from the Image Manager Interface (see 4.3.2). It accepts one argument, specifically the URL of the document and then queries the RDF repository in order to define the named entities which are cited on the specific document. Moreover, it calls the methods of Download Image class from the File

repository manager package in order to download the specific document. Except of the citations and the entities the Image downloader returns:

1. The URL normalized.
2. The width of the page.
3. The height of the page.
4. The total number of named entities (persons and places).
5. The date when the document got introduced by the user.
6. The URL of the document that contains the image.

Finally for each entity we return the coordinates on the image where the named entities are cited (see 4.4.4.1.1)

4.4.3.1.2 PDF Downloader

This downloader is called from the PDF Manager Interface (see 4.3.2).It accepts one argument specifically the URL of the document and then queries the RDF repository in order to define the named entities which are cited on the specific PDF document. Moreover it calls the methods of the Download PDF Class from the File repository manager package in order to download the specific document. Except of the citations and the entities the PDF downloader returns:

1. The URL normalized.
2. The number of pages of the PDF document.
3. The title of the PDF document.
4. The author of the PDF document
5. The total number of named entities (persons and places).
6. The date when the document got introduced by the user.

Finally for each entity we return the coordinates on the PDF document and the specific page where the named entities are cited (see 4.4.4.1.2)

4.4.3.1.3 Page Downloader

This downloader is called from the Page Manager Interface It accepts one argument specifically the URL of the document and then queries the RDF repository

in order to define the named entities which are cited on the specific web page. Moreover, it downloads web pages with static content calling the methods of the Download HTML page class of the file repository manager. Except of the citations and the entities the PDF downloader returns:

1. The URL normalized.
2. The title of the web page.
3. The total number of named entities (persons and places).
4. The date when the document got introduced by the user.

Finally for each entity it appends to a copy of the web document found in the file repository a sentence which contains the javaScript function (see 4.4.4.1.3) which selects the text that cites the named entity.

4.4.3.2 Search Components

These components consist of the JSP files which help the user search entities, citations and identity assumptions. They are an important part of the behavioral layer because with them a user can monitor the knowledge of other users, their identity assumptions e.t.c. These components receive calls from the document manager via Ajax requests and return the result of queries in order to reload the interface. These components return html source code because it is useful in order to change the state of the interface easily.

4.4.3.2.1 Entity Search

This type of search will be useful in order to start the historical research. In this type of search a user introduces the name of an entity and returns the attributes of the specific named entity. The entity search has 2 types as the types of named entities supported by the WhoWhereWhen platform, specifically persons and places. In addition the entity search guides the user in searching for the named entities and then finding the documents that cite the named entities (see 4.4.3.2.1).

The attributes for person entities which return the entity research are:

1. The multiple or alternative names of persons
2. Birth Date
3. Death Date
4. Identifiers in Authority files
5. Positions that the persons has had
6. The organizations where those position were.
7. Social relation with other persons
8. The user that knows and has introduced this persons' entity

The attributes for place entities which the entity research returns are:

1. The multiple or alternative names of a named entity
2. Identifiers on authority files
3. The latitude of place
4. The longitude of the place
5. The subordination of the place
6. The user that knows and has introduced the place's entity

As we described in 4.2 the knowledge about the entities comes from the users who expressed their knowledge of them. Finally, if a user has introduced two or more named entities which he/she identifies as same, we name these entities as merged (see 4.4.3.3.3).If two or more named entities are merged we consider these entities as one and we represent them on the interface as one entity.

4.4.3.2.2 Search Document which cites named entities

As we described above the entity search could be the first step of historical research. Also, there are cases that the entity search is not enough in order to support the historical research. In addition it will be useful for the researchers to find the documents that a named entity co-refers.

This type of search is the next step of entity search. If a user aims to find the documents that a named entity co-refers, firstly he/she should search a specific entity by name and then search the documents which refer to the named entity.

This search returns 3 types of documents which are the types supported by the WhoWhereWhen platform. For each type of document it returns the specific attributes:

- **Images**: The normalized URL of the document in order to present it graphically on the interface
- **PDF documents**. The normalized URL of the document in order to present it graphically on the interface and show the title of the document
- **HTML documents**: The normalized URL of the document, the selected text, the URL and the selected text which contained the page that cites the specific named entity.

If two or more named entities are merged this search returns the documents that refer to all the named entities which were merged.

4.4.3.2.3 Citation Search

As we have described, one of the main functionality of the WhoWhereWhen platform and CRIM model is the representation of co-reference that a named entity has in different documents. The citation search aims to help users find existing named entities which have already been introduced in the knowledge base and have already been cited in other documents. In this way the users avoid making a lot of duplicated named entities; in addition it is an easy way to present the documents that a named entity co-refers.

The citation search returns only the named entities that have been introduced from the specific user who uses the citation search. It returns the same attributes that the entity search returns. The next step after citation search is the submission by the users of their knowledge in order to define the documents that a named entity co-refers.

4.4.3.2.4 Identity Assumption Search

One of the most important functionality of the WWW platform is the Identity Assumptions that user expresses for the named entities. This functionality requires that the user can easily find the identity assumptions that other users have submitted for a specific named entity.

The identity assumptions search is called from the Document manager via Ajax requests and then returns the three types of identity assumptions (same , not same, possibly same) for the specific named entity.

4.4.3.3 Knowledge Submitter

The Knowledge submitter consists of JSP pages which enable the users submit their knowledge. It is one of the main components of this project because the knowledge that is introduced in RDF comes from the users exclusively. It is important for the WWW platform to provide the functionality and features to users regarding the introduction and submission of their knowledge for the named entities and their attributes, the citations on documents and finally the identity assumptions about different named entities.

These components are called from the document manager in which the users have inserted their knowledge and then introduced in the RDF sesame repository. The components of the Knowledge Submitter accept Ajax requests from the Document Manager Interface, convert the users' knowledge in RDF format in order to import it to the RDF repository. Although, there are cases that need checks of validity in order to ensure the consistency of the knowledge database. Below we describe the methods that we use in order for the users to import their knowledge.

4.4.3.3.1 Import of Named Entities

The basic term that we used in the CRIM model is the named Entity. In 3.2 we *defined as named entity an entity in which we can give a proper name.* The knowledge about the attributes of named entities is based exclusively on users' knowledge who express their theoretical interpretations for them. Moreover, we have defined that a named entity in CRIM is limited to person, place and event but the WhoWhereWhen enables the users to import knowledge only for people and places because their identification can be defined easier than events and the users import their descriptions and knowledge for persons and places in order to identify them with other named entities that have been introduced from other users. As we described in 4.2 the platform does not verify if the knowledge and the attributes which determine the named entities are true or if this knowledge is objectively acceptable from other actors. Moreover the platform does not aim to make interpretations or descriptions for the named entities. It provides information in order to help the user make identity assumptions for the named entities.

The JSP page which imports the user's knowledge and interpretation accepts an Ajax request and sends the data and information which are relative with the named entity. The data and the attributes are converted in RDF classes and properties of the CRIM model. The attributes of person are:

1. Multiple or alternative names of persons
2. Multiple identifiers of authority files
3. Birth date
4. Death Date
5. A position that he/she had in his/her lifetime
6. The group this position belongs to
7. The start and end date of the position
8. The social relation with other people

It also imports the date when the named entity is submitted and the user who knows and expresses his/her knowledge for that person entity.

The attributes for place are:

1. Multiple or alternative names of the place
2. Longitude
3. Latitude
4. Multiple Identifiers of authority files
5. Geopolitical subordination
6. The date that starts the geopolitical subordination
7. The date that ends the geopolitical subordination

As in person entity submission the system automatically imports the date of submission and the user who made it.

We note that all the attributes are optional during submission. Moreover, we note that the number of attributes that describe the named entity are limited because the platform does not aim at making interpretations for users but to help them to identify.

4.4.3.3.2 Citations

The WhoWhereWhen platform invokes a graphical functionality in order for the users to determine the area in a document where a named entity is cited. The citations demand the determination of two things:

1. Description of the named entity
2. Determination of the area in the document

The WhoWhereWhen interface guides the user in order to determine the area firstly and secondly to describe the Named entity. In 3.5 the classes and properties are described in order to describe the citation. The data which are needed for a citation are:

1. The URL of the web document
2. The data that specify the citation area on the document
3. The URI of the named Entity

4. The user which determined the area
5. The date of the submission.

A named entity could have a lot of citations in different documents. A way to determine the co-reference is to describe the area on the document once, then search the citation as described in 4.4.3.2.3 and finally submit the citation.

4.4.3.3.3 Identity Assumptions

The identity Assumptions as defined in 3.7 are the assumptions the user expresses for two named entities in order to determine their identity. The identification based on users knowledge is not strict, so we do not use terms as similarity , difference e.t.c. The users identity assumption has 3 types:

1. Same
2. Not Same
3. Possibly same.

It is obvious that a user can express only one type of identity assumption for two specific entities because he cannot assume that two entities are the same and not the same simultaneously. Finally, if a user has introduced two named entities in the knowledge base and he assumes that the entities are the same we define these entities as merged. The merged entities are considered as one named entity and represented as one entity in the WhoWhereWhen system.

4.4.3.3.4 Split

The split process is the reverse process of the identity Assumption. It is used when a user wants to separate two entities which are defined as merged. Also, if an entity splits, the entities which have been determined as “same”, “not same” and “possibly same” are determined as “possibly same” with the other entity. Algorithm 2 shows how the system splits a named entity:

Algorithm 2 Split Algorithm

```
SplitAlgorithm(SplitEntity,Entity)
    Q ← new RepositoryManager();  //It is used for the management of RDF repository
    SameAsEntitiesList    ← Q.getSameAsEntities(SplitEntity);
    NotSameAsEntitiesList ← Q.getNotSameAsEntities(SplitEntity);
    PossibleSameAsEntitiesList ← Q.getPossiblySameAsEntities(SplitEntity);
    For ( each SameEntity ∈ SameAsEntitiesList)
        AssumeitAsPossibleSame(SameEntity, Entity);
    endfor
    For (each NotSameEntity ∈ NotSameAsEntitiesList)
        AssumeitAsPossibleSame(NotSameEntity, Entity);
    endfor
    For (each PossibleSameEntity ∈ possibleSameAsEntitiesList)
        AssumeitAsPossibleSame(PossibleSameEntity, Entity);
    Endfor
```

4.4.3.3.5 Instance URIs and Contexts

Each instance of a class that is uploaded in the RDF repository must have a Unique URI. A way to create unique URIs and to monitor the sequence that entities are introduced on the knowledge database is the Date when the entity got submitted. The string that determines the date has the format dd_MM_yyyyHH_mm_ss. We use the underscore character because it is acceptable in the URI repository. Characters such as ":" will cause a problem on submission. Also, it is possible for two users to introduce named entities at the same time and due to this reason we use the username in the URI. Summarizing, the string which is the URI of the instance is a concatenation of the string of the date, the username and the name of the entity if it has been determined by the user.

Sesame offers the statement context mechanism and statements which are the quads (subject, predicate, object, context). Also, it provides methods in order to

remove instances using the Context of them. We use different context URI for each entity and citation because it is an easy way to remove the statement which is relative to it from knowledge base.

4.3.2 Data Flow

In previous sections we explained the architecture and the components of the WhoWhereWhen system. In order to understand better the functionality and the architecture of the platform, it will be useful to examine the data flow and the steps that are taken when a user interacts with the interface of the platform. The figure 4.3 represents the data flow of the WhoWhereWhen platform.

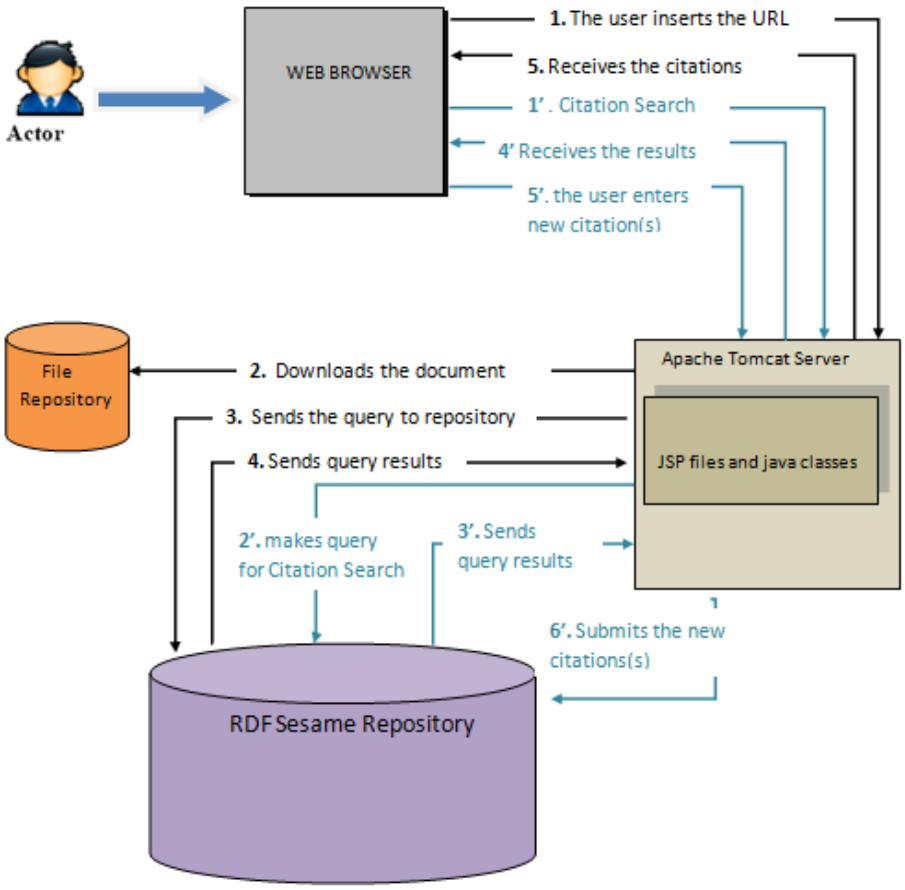


Figure 4. 3 Data flow of WhoWhereWhen

If we want to find a correlation between layers and shapes of the figure 4.3 the box which has the title “WEB Browser” refers to the representation layer which is the interface of the system. The box with title “JSP files and java classes” relates to the behavioral and repository layers and the cylinders are relevant to the File Repository and the RDF sesame repository which further correlates to the Storage Layer. The steps that a user follows in order to monitor the citations on the documents are:

1. The user inserts the URL of the document in order to open it and searches the entities which are referred on the document.
2. The File repository Manager downloads the document to the File repository.
3. The Repository manager sends a query to the knowledge base in order to determine the citations on the documents.
4. The document manager receives from the document downloader the citations which exist on the document.
5. The RDF sesame repository sends the results of the query

The steps that a user follows in order to introduce new knowledge are:

1’ The user determines the area on the document in order to cite a named entity and searches for existing citations.

2’ The component for the citation Search sends a query to the knowledge base.

3’ The RDF sesame repository sends the results of the query

4’ The document manager receives the results of the citation search.

5’ The user enters his own description about the named entity cited on the document.

6’ The Knowledge submitter submits the new citation and description for the named entity.

4.4.4 Representation Layer

This layer is responsible for the communication between the users and the WWW. It consists of the login and register page but also of the Document Manager which specialized in 3 types: Image, PDF and Page manager due to the different types

of documents supported by the WWW. Also, it is important to find solutions in order to manage documents and to determine specific areas on documents which cite the named entities. The challenge in the interface is the management and the representation of the different knowledge that the users have submitted to the knowledge base. Due to the difficulty in showing the different knowledge that users have it is important to develop a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of documents citations for named entities and associated identity assumptions.

4.4.4.1 Documents

It is useful to describe the documents that are supported by the WWW platform because it is an important part of the interface. The documents which are compatible with the WWW are images, PDF files and html pages because these types are the most popular in historical research. These documents are used in order for the users to determine the area where a named entity is referred. Below we describe how to determine areas in documents for the different types of documents that are supported, the methods we use in order to represent the documents and finally the decisions we took for the management of those documents.

4.4.4.1.1 Images

The Images in WWW commonly show persons or places but also in many cases they are documents which have been digitized in image type. The acceptable image types in WWW are the JPG[20] and PNG[21]. The platform uses the javax.swing.ImageIcon library in order to get information of the image such as width and height in order to keep the image unchanged.

The areas on an image are in rectangle shape. In order to determine the area it is enough to get the pixel coordinates only in two points. The points which are enough in order to determine the rectangle area of an image is the point A and B as shown in figure 4.4

[20] http://en.wikipedia.org/wiki/JPEG
[21] http://en.wikipedia.org/wiki/Portable_Network_Graphics

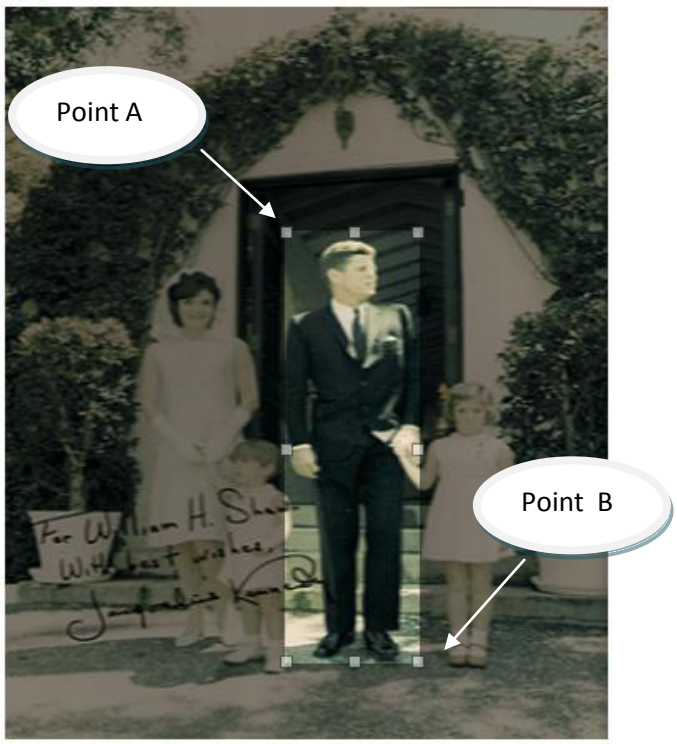


Figure 4. 4 Determination of Image Area

The WWW platform provides the functionality that the user determines the area. This functionality is named tagging. It highlights the image in order for the user to determine the rectangle area where the named entity is cited. In fig 4.4 we can see an example of use which highlights the specific area but also can change its shape easily.

4.4.4.1.2 PDF files

The PDF[22] documents commonly include text but also there are cases that include images or shapes. Nowadays, the PDF files are very common data resources in historical and scientific research. The platform uses the icepdf an open source Java PDF engine that can render, convert, or extract PDF content within any Java application or on a Web server. This engine extracts data which are relative with PDF files such as number of pages, author and title.

[22] http://en.wikipedia.org/wiki/Portable_Document_Format

The Area determination of a PDF file is a difficult process because as we described above the content of a PDF file has diversity. A PDF file may include images, text, diagrams or shapes. It is important to find a method which enables users select an area regardless of the content type of the document. An interesting solution to this problem was the usage of the JavaScript API[23] which provides functionality for PDF documents via the Acrobat software. However, this solution was rejected because it requires that each user of the system installs the software of Adobe Acrobat and the license in order to use this software is expensive. Moreover the WWW platform does not aim to enable users find documents and citations via the platform but to let them introduce the document they have already found on the WEB.

Due to the reasons which are described in the previous paragraph, we decided to develop a PDF viewer (fig 4.5) which converts the pages of the PDF files to PNG images. This way it is possible to determine the areas in documents regardless of the content type because we determine the areas as we do for images. Moreover, this method does not require the installation of software. The PDF viewer of the WWW is independent from other software and PDF viewers.

The area determination demands the pixel coordinates of Point A and Point B (fig 4.5) as with images but also demands the number of the page that includes the specific area.

The PDF viewer (fig. 4.5) demands a different type of caching than what we described in 4.4.2.3. It is difficult to store the PDF file in a PNG format because it requires a lot of time to convert all pages of the document to PNG . In addition it demands huge disk space. So, when a user presses the button for next page the PDF viewer creates the certain page in PNG format. The user can create a lot of pages which remain in the file repository in order to avoid the process of the page conversion if he wants to read them again. The images which are created from the pages are deleted, when the user closes or leaves the PDF manager interface. This process uses Ajax technology in order to call the class in the File repository manager which creates the PNG image of the specific page without refreshing the entire interface.

[23] http://www.adobe.com/devnet/acrobat/javascript.html

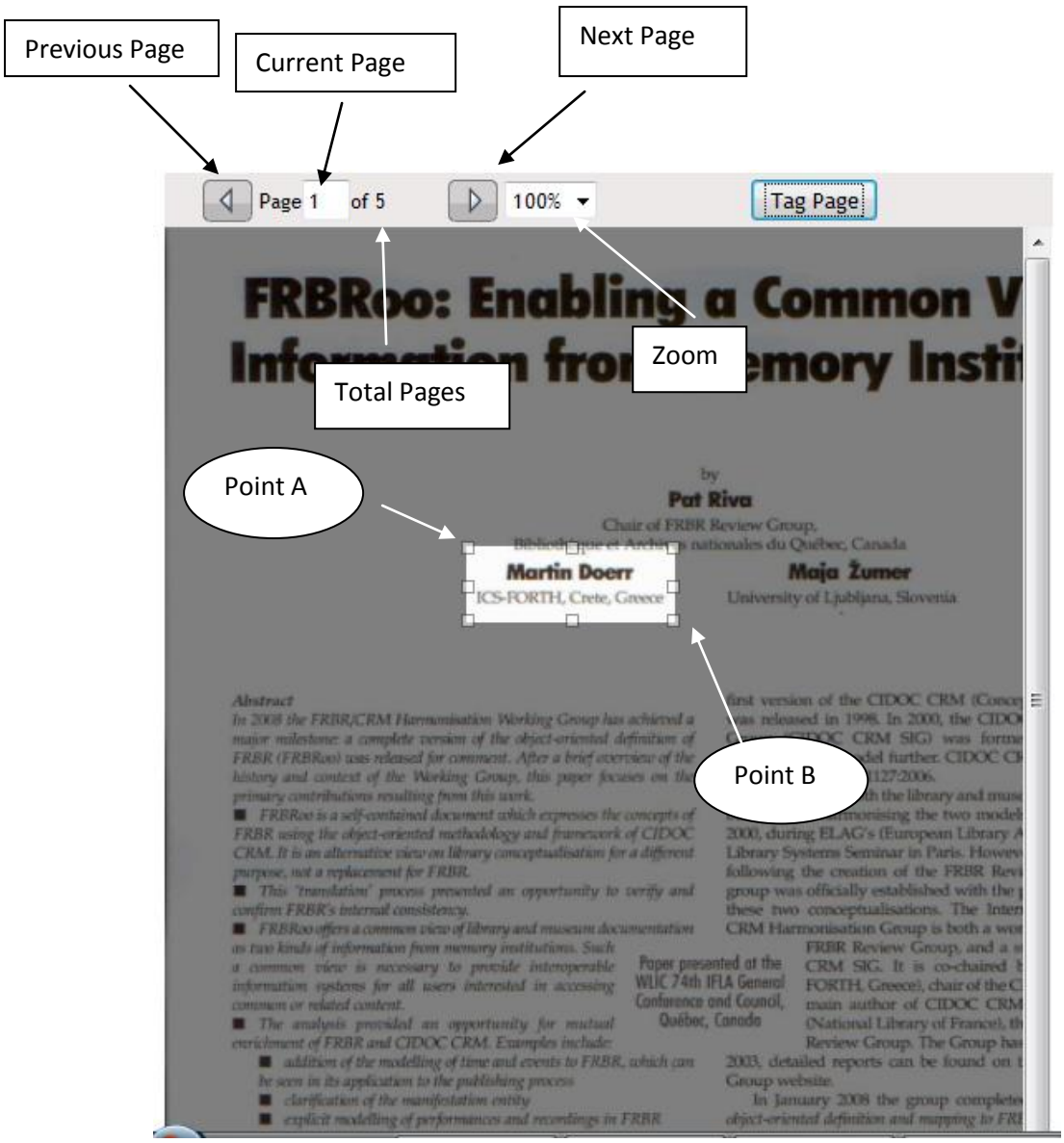


Figure 4. 5 WWW PDF viewer

4.4.4.1.3 HTML pages

The HTML pages commonly include text but also images and shapes. However, in an HTML page in comparison with the PDF document a user can easily extract the specific object using its URL. In HTML pages it was very difficult to define the specific area where a user selects text on the page. The text is not enough because there are cases that it is duplicated in the HTML document. It was important to find a method that makes it possible to get the metadata which define the exact area on an HTML page. It is easy to select the text in order to highlight it at runtime.

However, it is difficult to get the metadata which are necessary for the determination of the specific text area in order to highlight it at another time. The solution to this problem was the Range of a page. Using this you can select any part of an HTML document and do something with this information. The most common Range is the user selection.

Historical information is usually available on pages that do not change. The "Range" is partially robust against the change of a page. A Range is an arbitrary part of the content of an HTML document. A Range can start and end at any point, and the start and end point may even be the same (in which case you have an empty Range). The most common Range is a user text selection. As soon as the user has selected (part of) the text on an HTML page, this selection can be converted to a Range. However, one can also define Ranges programmatically.

If a user selects a text we will have two cases:

1)In this case the selected text is in one element(certainly paragraph Element).

```
<h4 id="entry1196"><a
        href="http://radar.oreilly.com/archives/2007/03/call_for_a_blog_1.html"
        class="external">Call for a Blogger's Code of Conduct</a></h4>

<p>Tim O'Reilly calls for a Blogger Code of Conduct. His proposals are:</p>

<ol>
        <li>Take responsibility not just for your own words, but for the
                comments you allow on your blog.</li>
        <li>Label your tolerance level for abusive comments.</li>
        <li>Consider eliminating anonymous comments.</li>
</ol>
```

Figure 4. 6 Example of selection

2)In this case the selected text is in two different Elements(In this case the element has different sorting).

```
<h4 id="entry1196"><a
        href="http://radar.oreilly.com/archives/2007/03/call_for_a_blog_1.html"
        class="external">Call for a Blogger's Code of Conduct</a></h4>

<p>Tim O'Reilly calls for a Blogger Code of Conduct. His proposals are:</p>

<ol>
        <li>Take responsibility not just for your own words, but for the
                comments you allow on your blog.</li>
        <li>Label your tolerance level for abusive comments.</li>
        <li>Consider eliminating anonymous comments.</li>
</ol>
```

Figure 4. 7 Selection Example

In the general case the selected text may include many different html-elements, as shown in the example below (The red text corresponds to the selected text):

```
<html>
    <body>
        <div>Heraklion is the biggest city of Crete</div>
        <div>Heraklion is the Capital of Crete</div>
    </body>
</html>
```

In order to define the selected text - in the general case - we must define the index of the character in the DOM element which specifies the start of the selected text in the specific DOM element. Moreover we must define for the specific DOM element, which is its position (child-index) in the parent DOM element. After that we must define the child-index of the parent DOM element in its parent's parent DOM element, all the way up to the root element. The same must be done for the end of the selected text.

In the example above to describe the selected text, we must:

- Start Character index is 0 at the first div Element of the first body Element in the document.

- End Character index at 30 in the second div Element of the first body Element in the document.

In order to highlight the selected text we hooked it in a span element which defines the style of the text. However it entails that the selected text be in only one element. If the selected area starts in an element and ends in another, the span element cannot be defined. The example below describes the problem:

In example 1 the selected text hooked in the span element included in the specific div element.

Example 1

<**div**>Heraklion <**span**>is the biggest city of Crete</**span**></**div**>

In example 2 the selected text starts in the first element and finishes in the second. In this example the definition of the span element is syntactically wrong and fails to highlight the text.

Example 2

<**div**>Heraklion is the <**span**>biggest city of Crete</**div**>

<**div**>Heraklion is the Capital</**span**> of Crete</**div**>

In cases such as the example 2 the WWW platform highlights the text until the end of the first element.

We clarify that the range can be determined only in areas of the document which are static and do not change dynamically. However, the highlight of the text which is based on the Range calls for code in JavaScript in order to determine the specific range and then to highlight it on the specific page. Also, an important security concept which is called *"same origin policy"*[24] permits scripts to run on pages originating from the same site in order to access each other's methods and properties with no

[24] https://developer.mozilla.org/en/Same_origin_policy_for_JavaScript

specific restrictions, preventing, though, access to most methods and properties across pages on different sites. The term "origin" is defined using:

1. the domain name
2. the application layer protocol
3. (in most browsers) the TCP port of the HTML document running the script.

Two resources are considered to be of the same origin if and only if all of the above values are exactly the same. The main reason that we use the caching process was the access problem that was created from the "same origin policy" scenario. In order to solve this problem the page manager downloads the html file of the web page and introduces JavaScript libraries' declarations in order to use them for the text selection and highlighting.

4.4.4.2 Example of use

It would be interesting before describing the interface of WWW to present a scenario of use in order to illustrate the utilities and the applications of the platform. The scenario that we choose is relative with the person Patrick Leigh Fermor[25]. The reasons that we choose this example are that Fermor has different occupations during his life and in many cases he has used nicknames because he was a soldier during the World War II. We present his biography shortly, more information for him is available in http://en.wikipedia.org/wiki/Patrick_Leigh_Fermor.

Sir Patrick 'Paddy' Michael Leigh Fermor DSO OBE (born 11 February 1915) is a British author, scholar and soldier, who played a prominent role behind the lines in the Battle of Crete during World War II. He is widely regarded as "Britain's greatest living travel writer". In World War II Leigh Fermor fought in Crete and mainland Greece. During the German occupation, he returned to Crete three times, once by parachute. He was one of a small number of Special Operations Executive (SOE) officers posted to organize the island's resistance to German occupation.

[25] http://en.wikipedia.org/wiki/Patrick_Leigh_Fermor

Disguised as a shepherd and nicknamed ***Michalis*** **or** ***Filedem***, he lived for over two years in the mountains. With British officer Captain Bill Stanley Moss MC as his second in command, Leigh Fermor led the party that in 1944 captured and evacuated the German Commander, General Heinrich Kreipe. The Cretans commemorate Kreipe's abduction near Archanes a village of Crete island. Finally, Fermor has written a lot of books mainly about travelling which have received quite positive reviews from critics.

In this example we have 3 different users which express their descriptions for Fermor based on their distinctly different knowledge for him. The first user has the username Gregory. He knows Patrick Leigh Fermor as a travel writer. The documents that he has found are relative with this occupation of Fermor. He has firstly found a PDF file which is an essay review for the writings of Fermor and an image which shows him in a book cover which he has written. In fig 4.8 we can see the citations and the area determination of the document that user Gregory has defined.

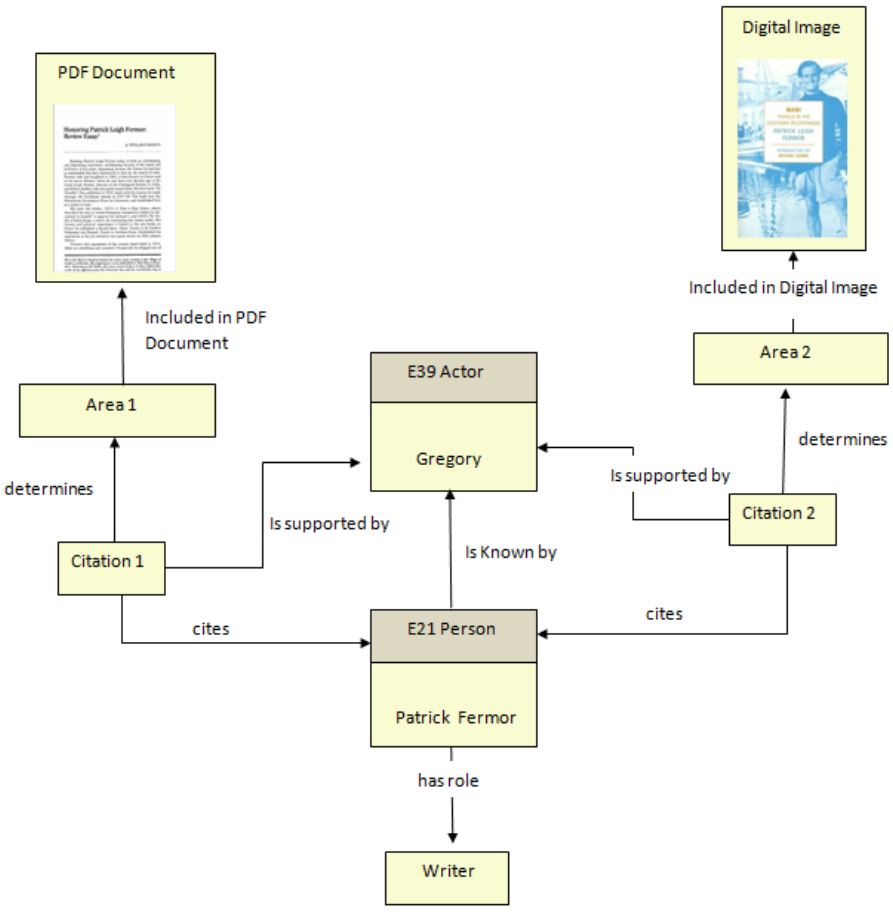


Figure 4. 8 The named entities and documents which supported by user Gregory

The user Martin knows Patrick Leigh Fermor as an officer who led the party that in 1944 kidnapped the German Commander, General Heinrich Kreipe . The photos that are cited by this user come from this event. In fig 4.9 we can see the documents that refer to Patrick Leigh Fermor and the determinations of areas that Martin has defined.

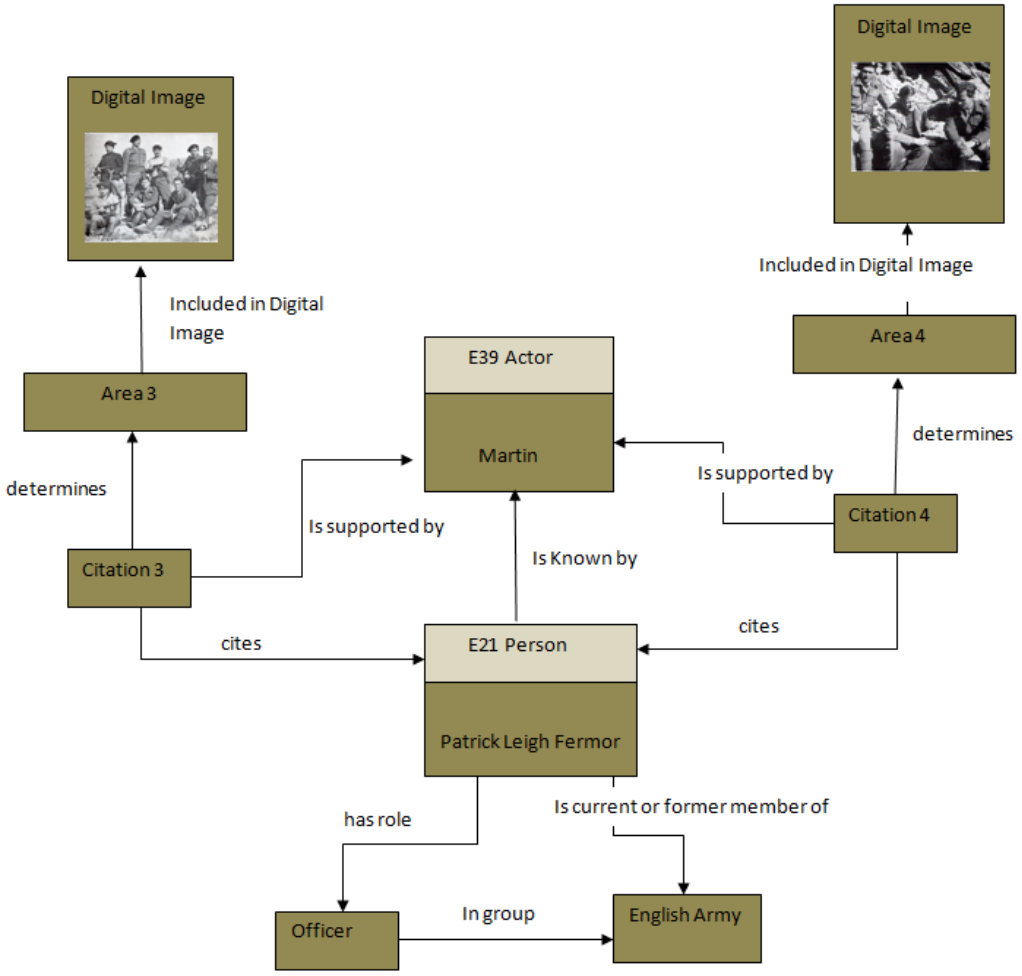


Figure 4. 9 The named entities and documents which supported by user Martin

In addition Martin in the second image has determined areas on the document that cites other persons. Specifically, in this document Heinrich Kreipe in the middle and Stanley Moss[26] in the left part of the picture are cited .

[26] http://en.wikipedia.org/wiki/W._Stanley_Moss

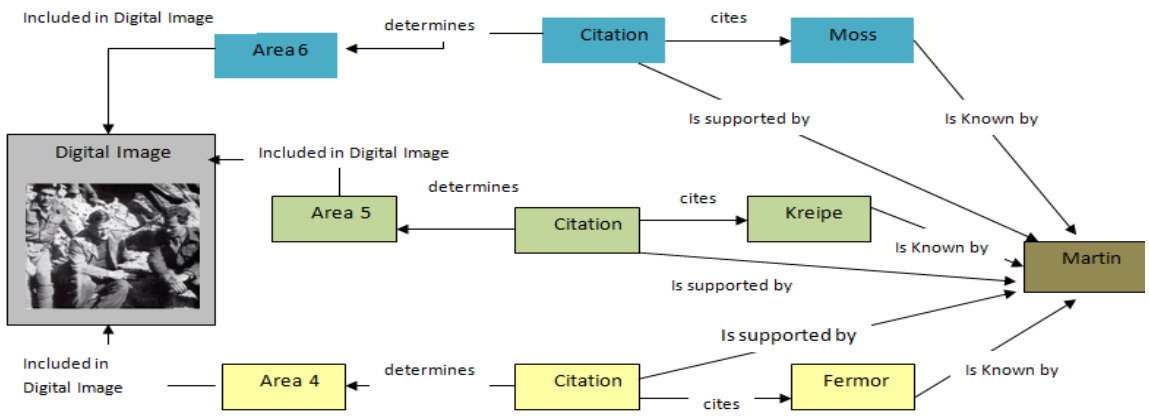


Figure 4. 10 Determination of Areas in Document Image

The last user has the username Valantis. Valantis knows the nicknames of Fermor and notes them via an html document. The knowledge of this user comes from a specific document which provides information for Fermor as a soldier and as a writer. By noting the nicknames of Fermor he achieves in making his own identity assumptions for him. In figure 4.11 we can see the determination of reference in an HTML document.

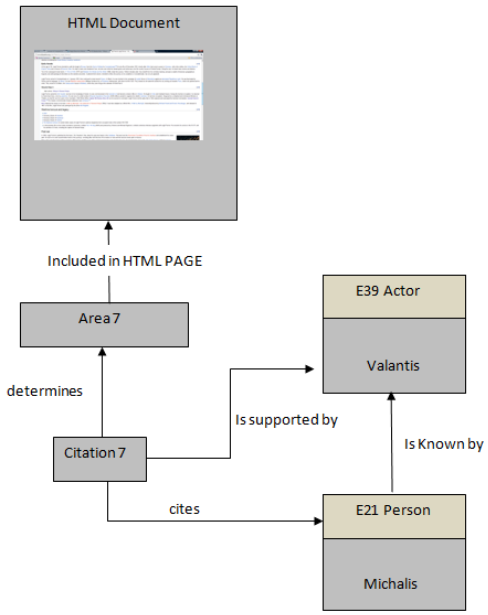


Figure 4. 11 The named entity and document which supported by user Valantis

The next step after the description of named entities and the determination of citations and areas on the documents is the making by the user of his own identity assumptions for the named entities that other users have described. In fig 4.12 the identity assumptions with green color determine two persons as same, with red color as not same and with blue color as possibly same. The identity assumptions are based on the knowledge of users and for this reason we define that the identity is based on users' assumptions and does not define strict similarities or differences.

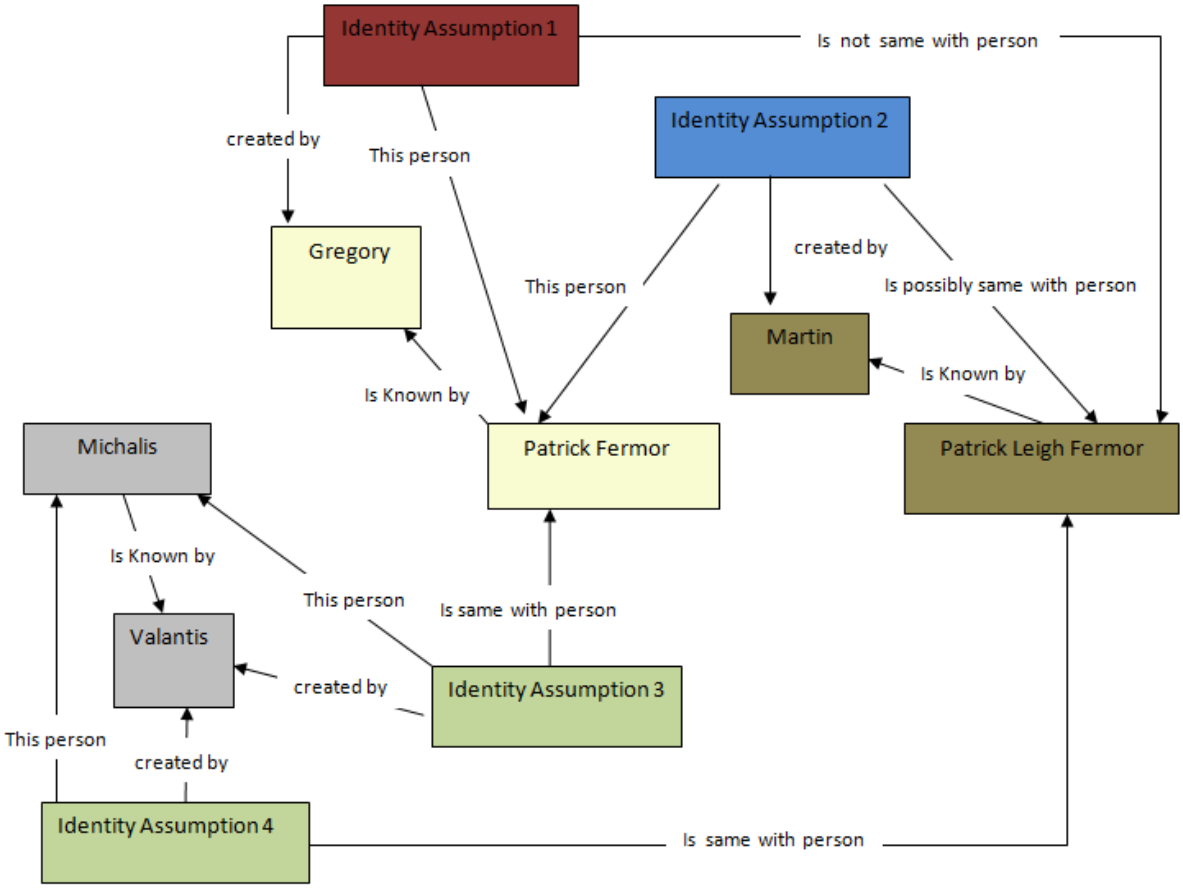


Figure 4. 12 Identity Assumptions of users

Below we describe the identity assumptions in fig 4.12:

- Identity Assumption 1 (created by Gregory)

  In this assumption the user Gregory assumes that Patrick Fermor(is known by user Gregory) is **not same** with person Patrick Leigh Fermor (is known by user Martin)

- Identity Assumption 2 (created by Martin)

  In this assumption the user Martin assumes that Patrick Fermor (is known by user Gregory) is **possibly same** with person Patrick Leigh Fermor (is known by user Martin)

- Identity Assumption 3 (created by Valantis)

  In this assumption the user Valantis assumes that Michalis (is known by user Valantis) is **same** with person Patrick Fermor (is known by user Gregory)

- Identity Assumption 4 (created by Valantis)

  In this assumption the user Valantis assumes that Michalis (is known by user Valantis) is **same** with person Patrick Leigh Fermor (is known by user Gregory)

  We use part of this example in order to demonstrate the interface of the WWW system.

4.4.4.3 User Interface – Document Manager

In this section we describe the user interface of WWW. The interface of WWW is developed in order to make it possible for the users to have other windows open such as pages or images while using WWW, in order to use them during their research. Furthermore, it is important to support users or groups of users to make their descriptions for the named entities and to determine the citations on documents by defining the specific area on them that mention the named entity. Moreover, it is important to search the named entities and the documents that refer to them. The co-reference of a named entity on different documents demands an interface that helps the user easily monitor the citations and the different documents that these named entities are referred to. Moreover, it must enable the user to monitor the state of different terms such as citations and named entities easily and efficiently by using techniques such as the highlighting of them with different colors. Finally, the challenge was to develop a user friendly and intuitive interface that guides the user in a natural way to distinguish the notions of document citations and associated identity assumptions.

The document manager is specialized in 3 types (Image, PDF, Page manager) due to the different types of the documents that the user may open and it is divided in 6 parts. We used tabs in order to divide the interface. Each tab provides different types of knowledge to the user in order to monitor easily the state of the knowledge base. The parts of the interface are:

1. Document Search
2. Document View
3. Make Citation
4. Entity View
5. Entity Search
6. Same As editor

The different tabs which are used in the WWW interface are a useful technique in order to represent different aspects of the knowledge. It is important that a lot of queries are run in order to collect the required data for the interface. So, during the research a user may search for multiple entities or other terms. It is important in specific processes such as searching to display information on specific areas without changing the content of all the tabs of the interface. In order to avoid a delay in refreshing the entire interface we use Ajax which enables the retrieval of data from the server asynchronously in the background in order to dynamically display the requested data on a specific area of the same page.

The challenge in using tabs was the transition of user to different types of knowledge. In figure 4.13 the user model of the WWW platform represents the transitions that a user makes in order to monitor the knowledge that interface represents. Firstly the system solicits the user to make a document or entity search in order to open a document. The user, then, can monitor the citations of named entities on the document or select another tab in order to use another function of the WWW. The shapes in grey color represent decisions or selections that a user makes. The red color represents processes on the document search tab, the green processes on the document view tab, the blue processes on the make citation tab, the yellow processes on the entity view tab, the purple processes on the Entity Search tab and finally the orange represents processes on the same as editor tab. The arrows represent the actions of the user.

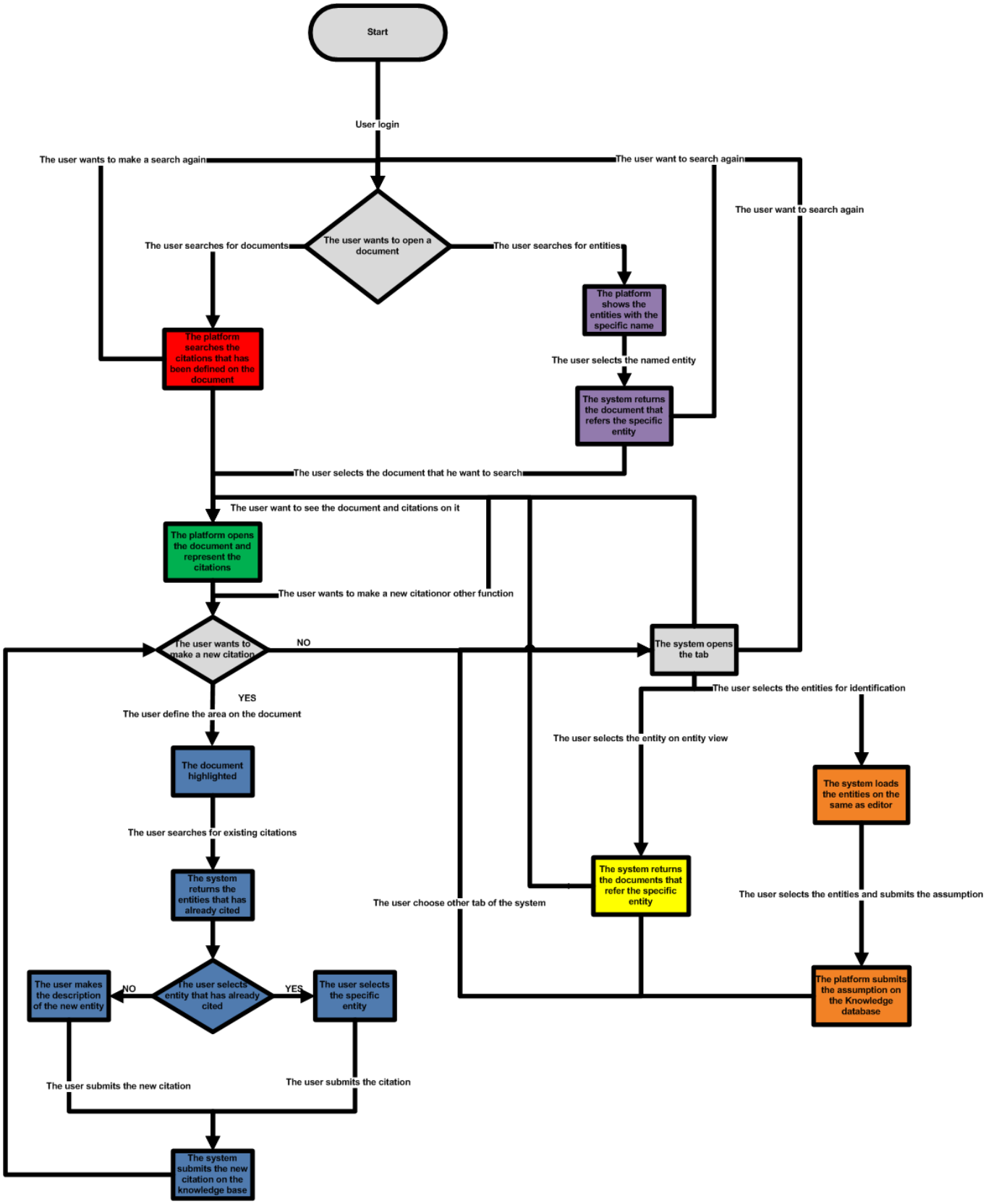


Figure 4. 13 User model of WWW

The figure 4.14 shows the login page where the user can start using the WhoWhereWhen. In WWW the users or groups of them determine their knowledge

for citations and named entities but also it is important to keep their username in each record in order to know the origin of the knowledge.

Figure 4. 14 Login page of WhoWhereWhen

The description of the components that follows uses the example that we described in section 4.4.4.2 in order to outline the functionalities and the components of the WWW system. We explain how the system represents the different knowledge that 3 users express about Patrick Leigh Fermor and how the users distinguish the notions of document citations and associated identity assumptions.

4.4.4.3.1 Document Search

The first part of the user interface is the Document Search tab. The documents as described in 4.4.4.1 are an important part of the WWW because they are used in order to determine the areas in which entities are referred. As we can see in figure 4.15 the user inserts the URL of the document, then the WWW system opens the document and searches for the referred entities within the specific document. On the left part of the interface it provides metadata to the users for the document in order to help them in their research. The interface of the WWW is adapted to different kinds

of documents depending on their size and functionality. If a user wants to open a PDF file, loads the PDF manager, if he wants to open an image loads the image manager and finally for HTML pages loads the page manager. In section 4.4.4.3.2 and 4.4.4.3.3 we described the different functionalities that the interface provide for each type of document.

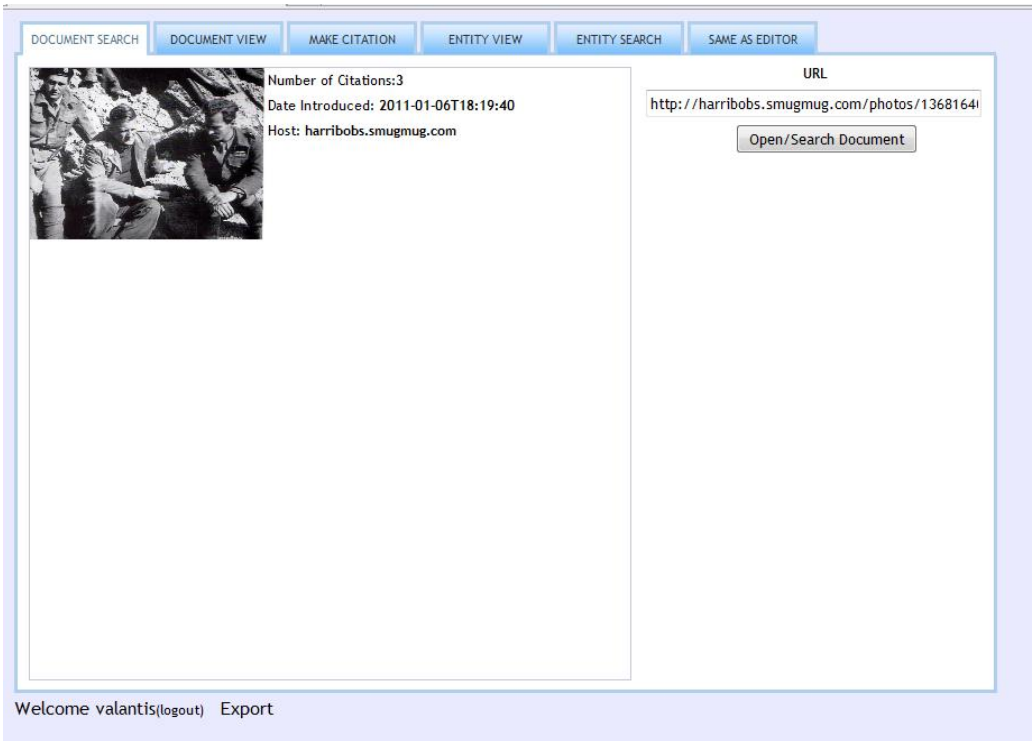


Figure 4. 15 Document Search

4.4.4.3.2 Document View

The second tab of the interface shows to the user the citations of the documents. In figures 4.16 , 4.17 and 4.18 we can see the different representations of the documents. On the left part of the interface we can see the documents that cite the names entities. On the right part of the interface metadata are provided for the documents and the descriptions of the named entities as they were described by the users. Each entity has a different color for its description being same with the color of the rectangle that determines the area on the document that refers to the specific entity. The highlighted area on documents differs depending on their type. For images

(fig 4.14) we use rectangles with different colors in order to determine the specific area. The figure 4.14 is a representation of the example in fig 4.16

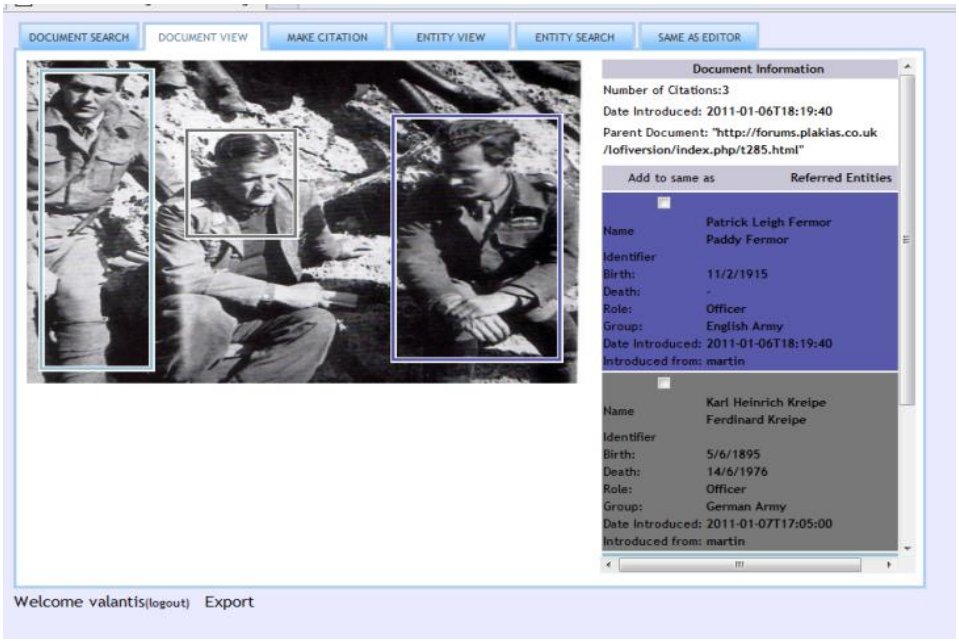


Figure 4. 16 Document View in images

In PDF files (figure 4.17) the highlighted part of the document is made in the same way as with image files. It requires the specific page where this citation exists.

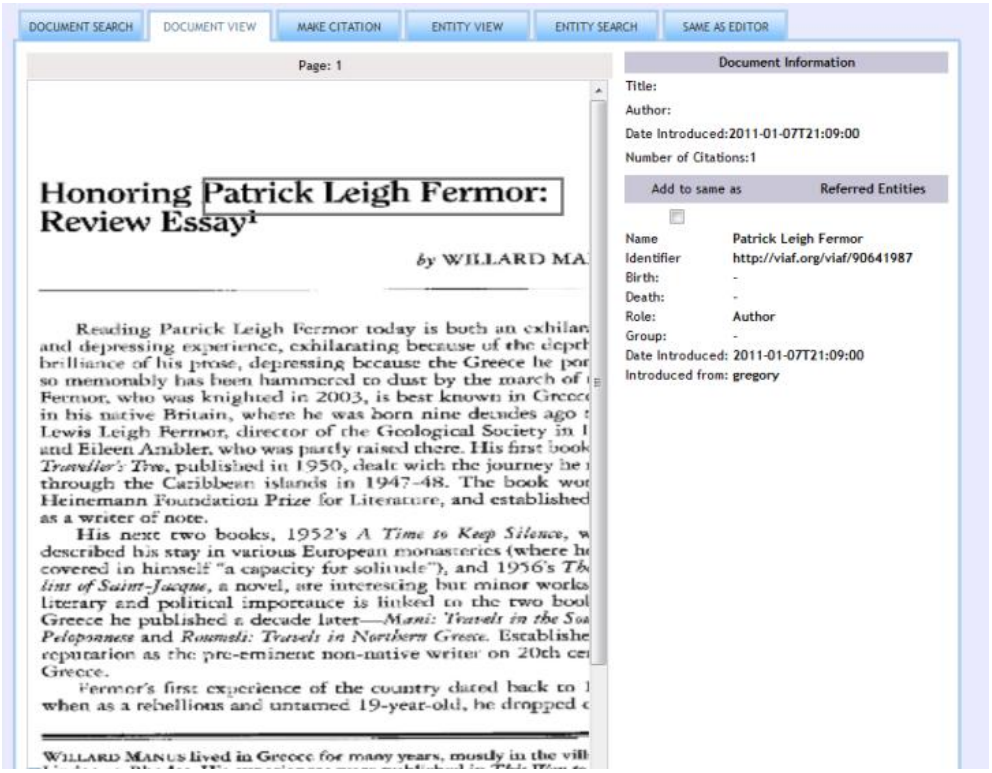


Figure 4. 17 Document View in PDF files

The highlighted area on html documents differs from the other two kinds of documents. The WWW interface highlights the area which is defined by the DOM tree as described in 4.4.4.1.3. It highlights the text with the same color as that of the entity's description shown on the right part of the interface. The figure 4.18 represents the example on figure 4.11 in which the user Valantis determines the area on the HTML document that the Entity's description is beside the document in which is cited.

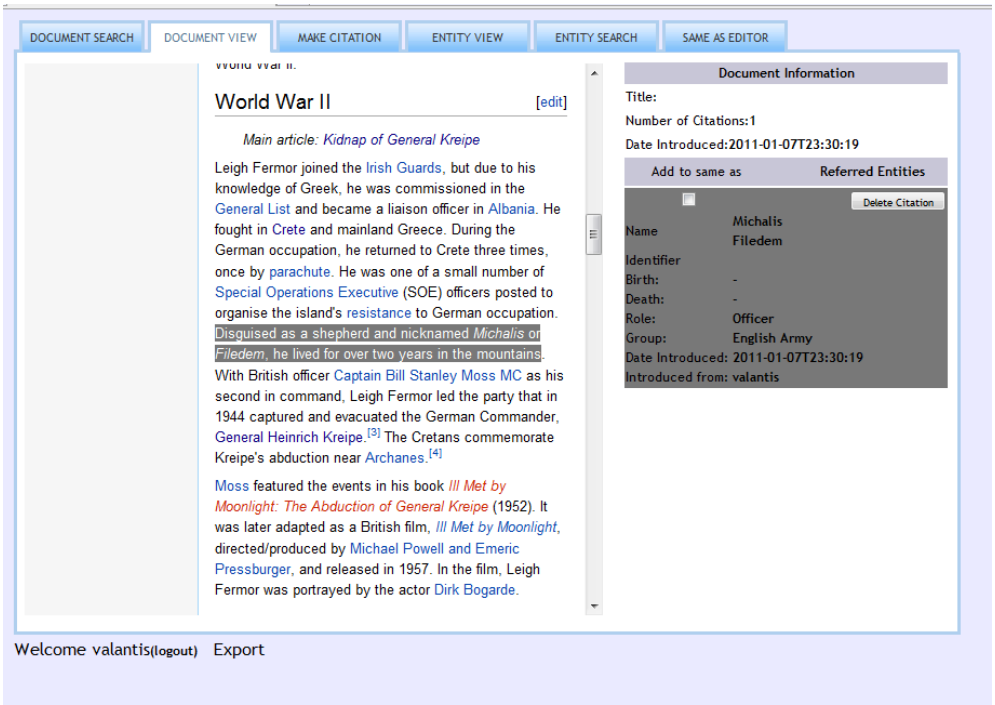


Figure 4. 18 Document View in HTML pages

4.4.4.3.3 Make Citation

The third part of the document manager enables the users insert their knowledge on the knowledge DB. This tab provides the functionality that the user determines the specific area on the document that cites the named entity, makes the descriptions about named entities and determines the co-reference of them.

The user determines the area on the documents firstly and then searches if he has entered the specific entity again. If he wants to enter a new named entity he

presses the appropriate button, the form appears and he introduces his description as shown in figure 4.19. Then the user determines the area and searches for existing citations. In fig 4.20 the user Martin fills the form in order to enter his description of the entity based on his knowledge about Moss. The selection area process differs accordingly to the other kind of documents as described in 4.4.4.1.

Figure 4. 19 Area determination on the document

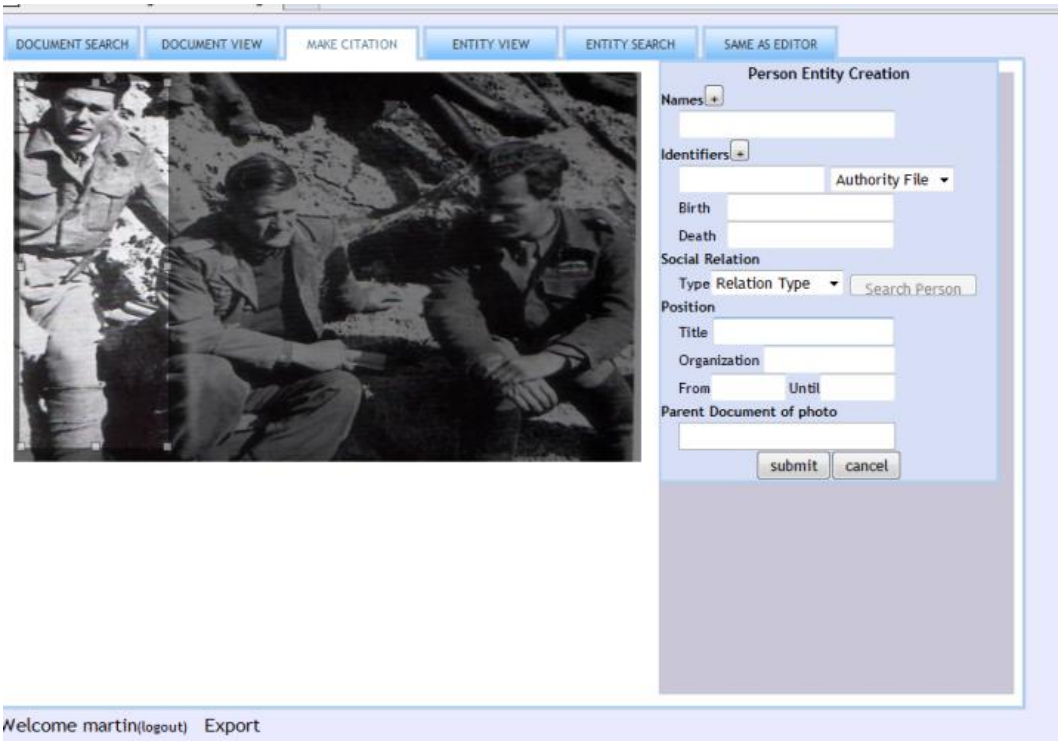


Figure 4. 20 The user enters his knowledge on KB

The user Martin has already made a citation about Patrick Leigh Fermor. He wants to determine that the second image as shown in fig 4.21 cites the same Fermor who has already been described. In this case the user Martin determines the area and then he searches for existing citations which he has defined. If Martin has already made citations for the specific person the WWW shows the descriptions about him. Then if Martin agrees with the specific determination selects the appropriate button and then the WWW goes into the named entity view (figure 4.22). Finally the user presses the submit button and enters the citation. The citation search returns the entities that the specific user has already entered in order to determine the co-reference between documents for the particular entity. It does not allow users to search for entities that other users have described.

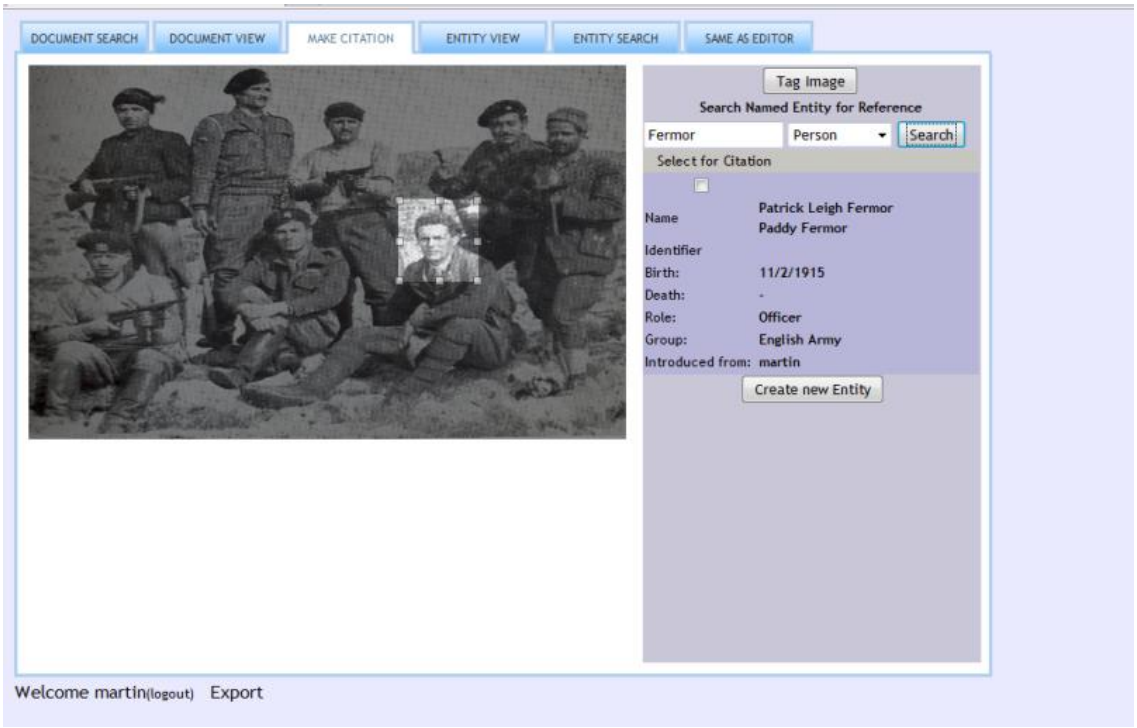


Figure 4. 21 Citation search

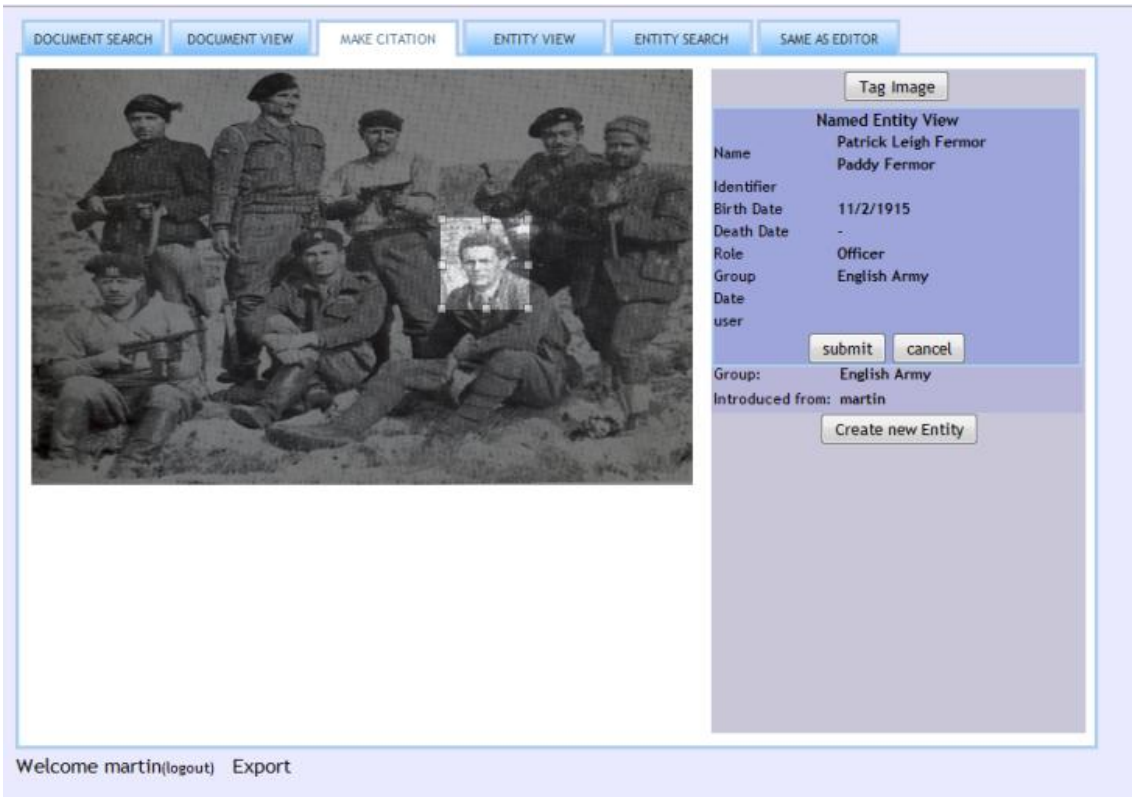


Figure 4. 22 Co-reference determination

4.4.4.3.4 Entity View

The entity view is the fourth tab of the WWW interface which aims to help the user in monitoring the different documents that refer to a specific named entity. The tab of entity view shows on the right part of the interface the entities that are cited in the document which is open at that time (information regarding the document itself and also whether the document is "visible" by the following tabs: document search, document view and make citation). The user presses the descriptions of named entities on the right part and then on the left the WWW shows the documents where this entity is referred. In figure 4.23 we can see that Patrick Leigh Fermor described by the user Martin is mentioned in two images as described in figure 4.9. In figure 4.24 the user Gregory has made another description of Patrick Leigh Fermor because he knows him as a writer. Gregory has determined that this entity is referred to an image and a PDF file. This example represents the co-reference that is described in figure 4.8. Finally, the interface of WWW allows the users to monitor the other documents that named entities of the open document were found to be cited.

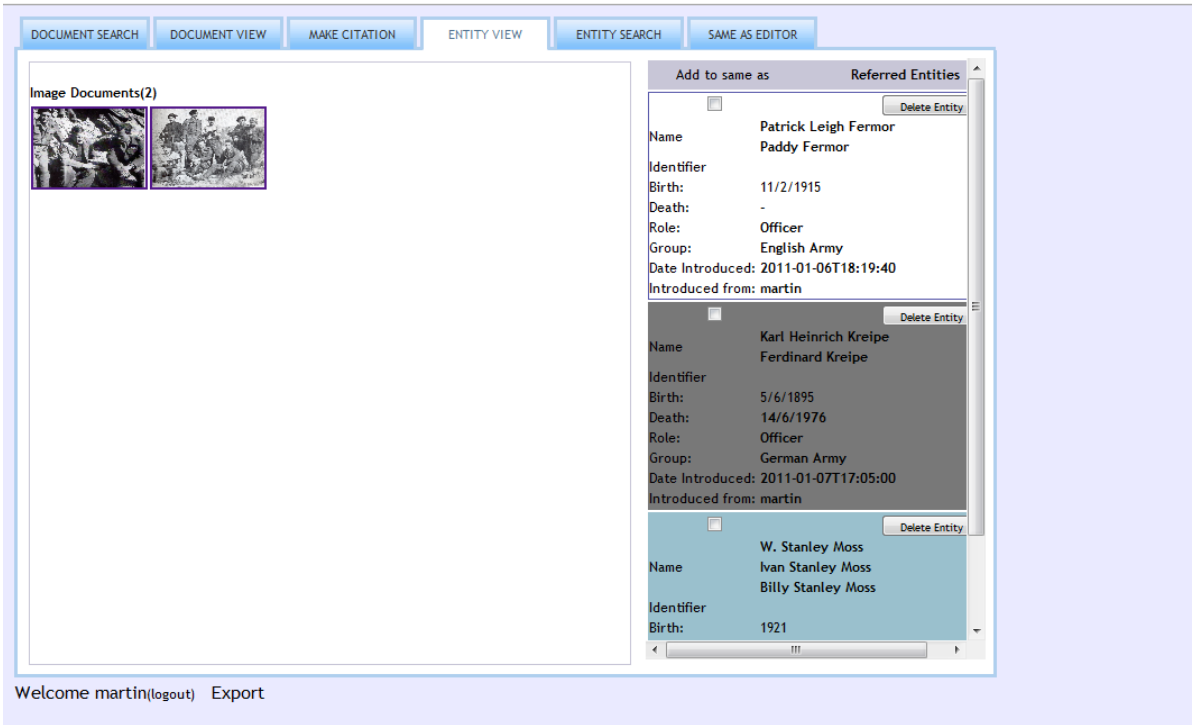


Figure 4. 23 Entity View

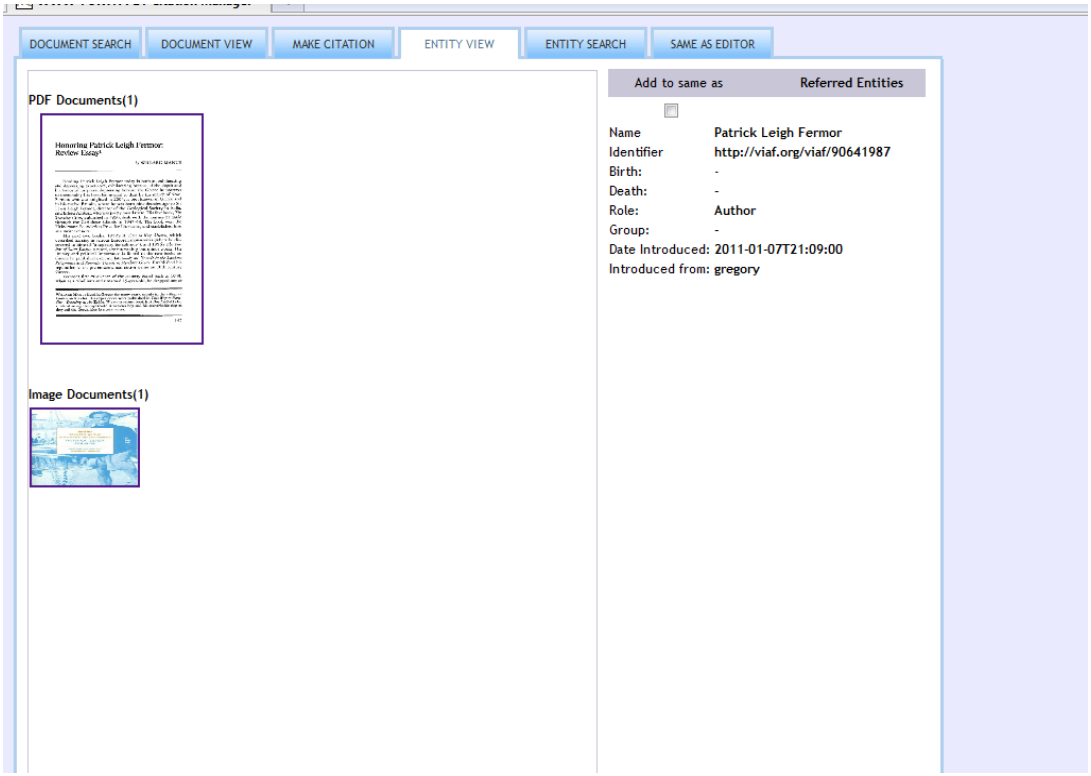


Figure 4. 24 Entity View

4.4.4.3.5 Entity Search

This part of the WWW enables the user to search entities which are kept in the Knowledge DB. Sometimes the historical analysis starts from this tab because it is an easy way for the users to monitor the state of Knowledge which is kept in the Database. The figure 4.25 represents the results that the search process returns when the users search for the entities which have the name “Fermor”. On the right part of the interface the search process shows the entities which contain the name that is used as a key for the search. Then the user selects the entity that he wants to monitor and the documents that this entity is referred at. The figure 4.25 represents the state of the knowledge DB as defined in the example in section 4.4.4.2. In figure 4.25 we can see that the user Martin describes Patrick Fermor as an officer of the English army and he assumes that this person is referred to two images which appear on the left part of the interface. Finally, the figure 4.26 shows that user Gregory knows Fermor as an author and he assumes that Fermor is referred to an image and a PDF file.

Figure 4. 25 Entity Search

Figure 4. 26 Entity Search

4.4.4.3.6 Same As editor

The last tab of the interface enables the users to make their own identity assumptions and search the existing assumptions which are associated with a specific entity and have been described by other users. The user inserts the entities on the same as editor tab selecting them by their descriptions on Document View or Entity View. The user selects two named entities in order to describe his own identity assumption. The user can define two named entities as *"Same", "NOT Same" and "Possibly Same"*. The user presses the submit button in order to enter the assumption on the Knowledge Database. In fig 4.27 the user Martin determines two named entities as *"Possibly same" as* described in the figure 4.12.

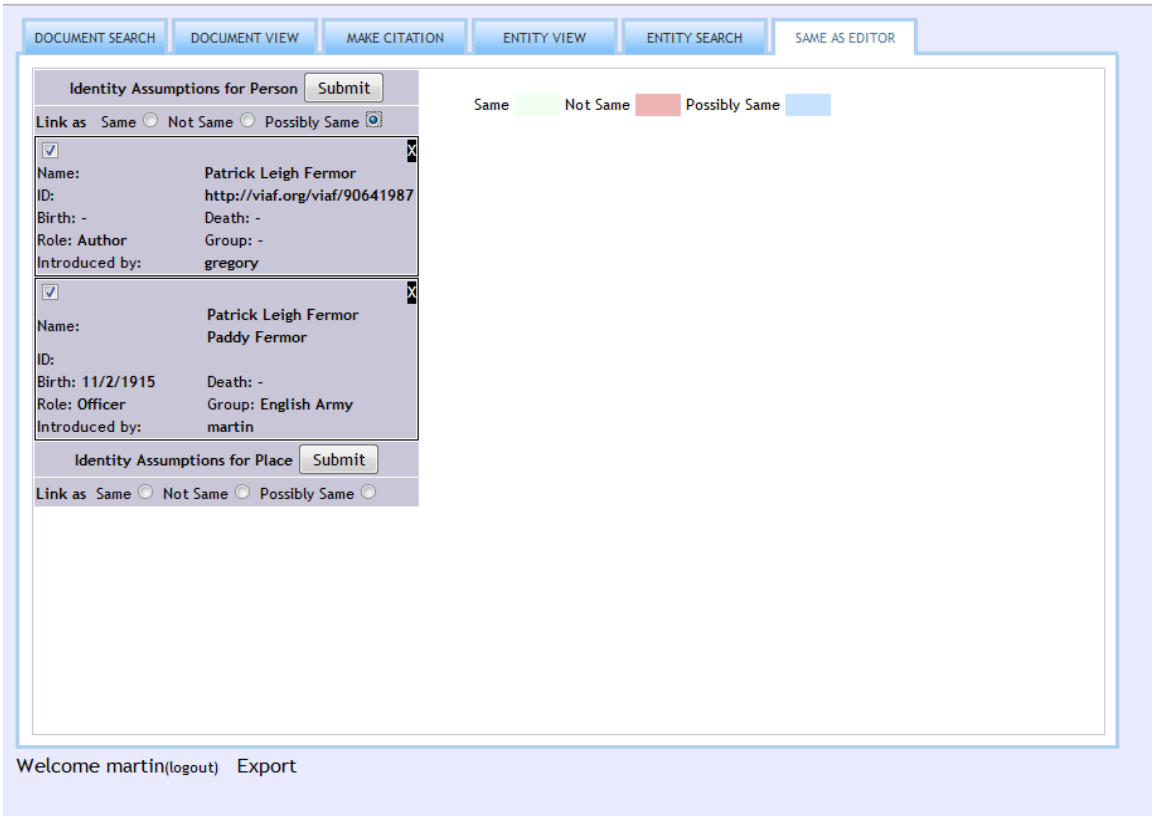


Figure 4. 27 The user Martin makes his own identity assumptions for two different named entities

It is important for the users to easily understand the type of identity that the different entities have. In figure 4.28 we can see how the WWW represents the

identity assumptions which are associated with a specific named entity. On the left part there are the descriptions of the entities we want to identify or search for their identity assumptions. The user presses the description of the named entity and on the right part of the interface the identity assumptions are presented. The interface uses green color for the "same as", red for the "not same as" and blue for the "possibly same as" identity assumption. In figure 4.28 Patrick Leigh Fermor who is described by the user Gregory is not same(assumption created by Gregory) and possibly same (assumption created by martin) with Fermor who is an officer. Finally, he is the same with Michalis (assumption created by Valantis). The figure 4.28 represents part of the example which is described in figure 4.12.

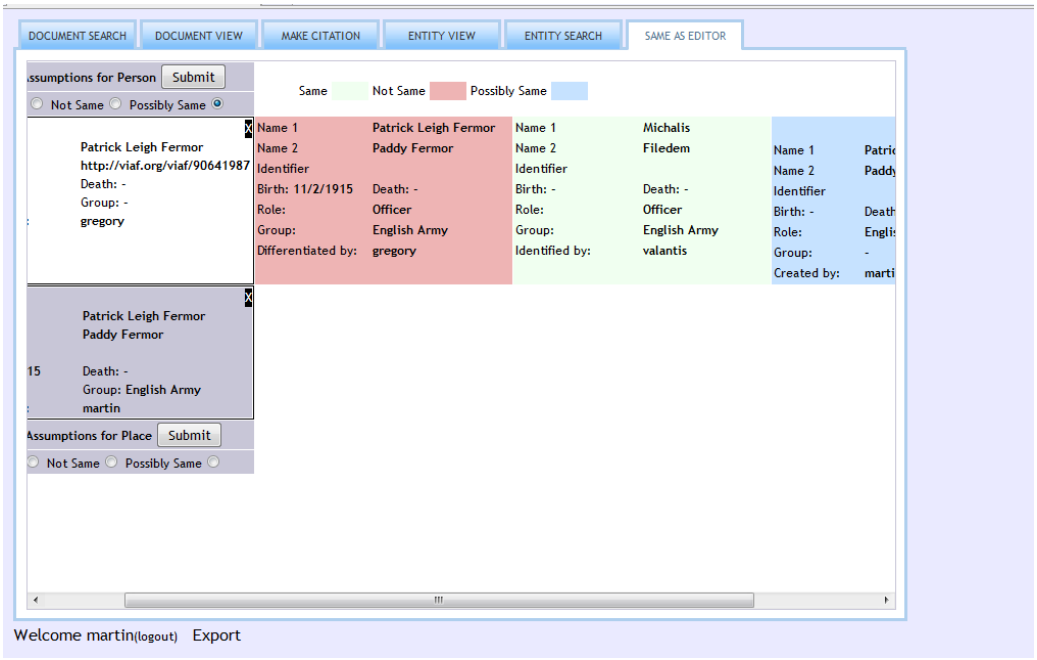


Figure 4. 28 The users search for assumptions which is associates with this entity

The figure 4.29 graphically represents all of the identity assumptions described in figure 4.11

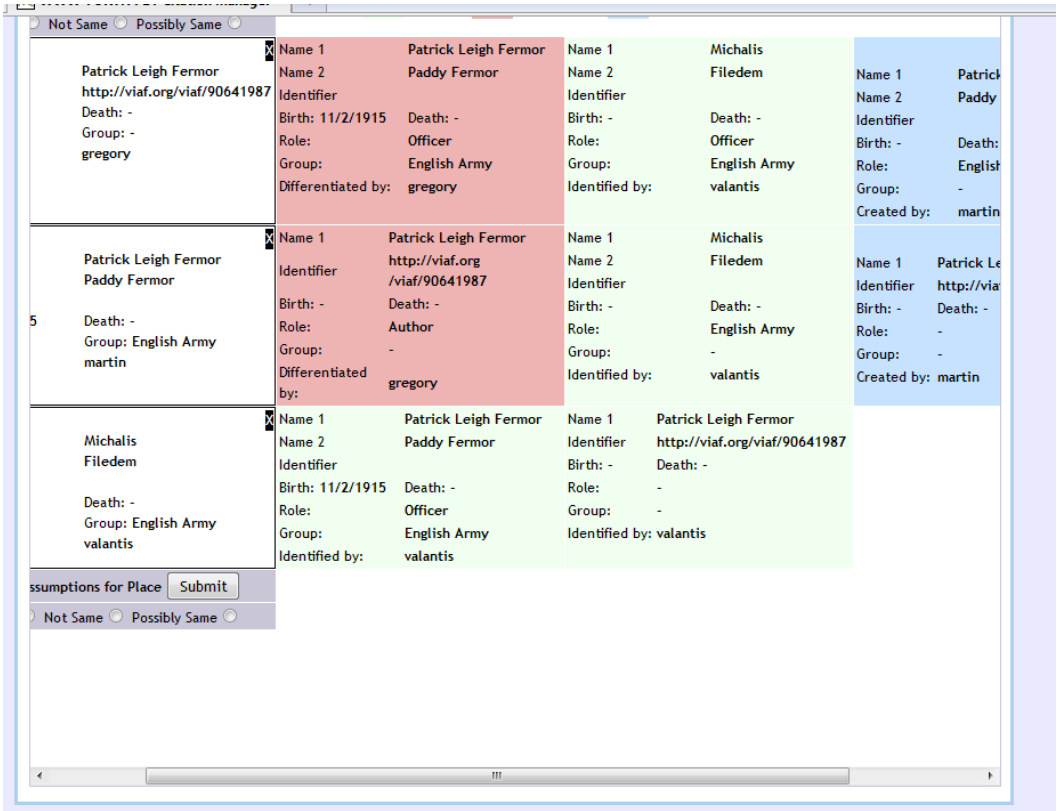


Figure 4. 29 The graphical representation of identity assumptions.

4.4.4.4 Programming Languages and techniques that were used on the WWW platform

The programming languages and the techniques which have been used on the WWW platform are:

- **Java**

  It is used for the communication with the File Repository or RDF repository.

- **JSP**

  It is used on the WWW for the communication between the representation layer and the lower layers. The programs that are developed in this language are called via AJAX requests.

- **JavaScript**

  It is used in order to dynamically change the content of the WWW platform.

- **CSS**

  It is a style sheet language used to describe the presentation semantics (the look and formatting) of a document written in a markup language such as HTML.

- **AJAX** (Asynchronous JavaScript and XML)

  The interface of WWW provides huge amount of information on different tabs. It is important to change only specific parts of the interface when we enter our knowledge or search for entities and documents without refreshing the interface totally. We use AJAX in order to change parts of the interface in order to avoid delays refreshing the entire interface.

Chapter 5

5 Conclusions and future work

In this thesis we describe a model (chapter 3) which enables the users represent their knowledge about the named entities, the citations and the identity assumptions about them. Also, it allows users to correlate the references of entities in a variety of documents of different types. It enables single or groups of users represent their descriptions for named entities. These descriptions work supportively in order help other users make their own identity assumptions for the named entities. Furthermore, this model can describe co-references of named entities whose descriptions and determinations are based on real events and conditions. It is of major importance to this model to define the distinctions of different knowledge and perceptions that the users provide for the named entities, their own identity assumptions along with the citations the users correlate. In addition, it enables the users to determine their identity assumptions for different named entities. Finally, this model introduces the distinction between the classification and categorization by representing the identity determinations based on the assumptions of different users.

We developed WhoWhereWhen (described in Chapter 4), a web based information system which enables the users make their own identity assumptions for the named entities. This system differs from others because the knowledge that it manages and dissipates is based upon the user's individual knowledge exclusively and differs from automatic named entity recognition methodologies which cannot work perfectly in all cases and demand enough complete and comparable data. Moreover, methods such as folksonomies aim at classifying and categorizing documents and not at determining the identity of different named entities. Moreover, other systems do not provide a user-friendly interface that presents the co-reference of named entities in different documents. The WWW (WhoWhereWhen) platform:

- Represents the co-reference of named entities in different documents.
- Provides descriptions of named entities which are based on users' knowledge.

- Does not aim at creating descriptions for the named entities but to use the available ones only in order to help users make their own identity assumptions
- Supports different types of documents
- Supports multiple users to work collaboratively
- Represents the citations of documents and associates them with the descriptions of entities graphically
- Determines the areas in a document which cites a named entity
- Guides the users to distinguish the notions of document citations and associated identity assumptions
- Enables users make their own identity assumptions for named entities based on their knowledge
- Provides the users with search mechanisms that can monitor the state of the knowledge Database. Users can search descriptions for named entities, citations, documents and identity assumptions of users
- Uses ideas of WEB 2.0 which enables users interact with the platform and the documents.
- Uses technologies of semantic web in order to represent the user descriptions and identity assumptions

We believe that the WhoWhereWhen can be used in but not limited to historical analysis which is based on documents that are available on the web. The researcher can easily determine the areas in a document that a named entity is referred and helps him/her make his/her own identity assumptions.

In the future we are planning to extend the CRIM in order to support other kinds of named entities and other types of documents. Moreover, we aim at extending the interface of the WWW in order to support other types of documents such as doc files, google maps e.t.c. and the determination of the areas in the specific documents. In addition, we aim at finding another way to cache the web documents because the file repository demands are quite significant and require huge disk space. The solution for this problem is the Memento project as described on 4.4.2.3. When this project will be more mature it can lead on the rescission of the file repository and of the caching process. Finally, it allows using documents that were retrieved in the past and now are retained only in organizations which provide web caching. Finally approaches and

mechanisms of automatic Named Entity Recognition can work supportively in the WhoWhereWhen platform in order to support the researchers in historical analysis.

Bibliography

[1] **D. Nadeu, S. Sekine**. "A survey of named entity recognition and classification ", In Lingvisticae Investigationes, Volume 30,pages 3-26, 2007

[2] **Alisera Mansouri, Lily Suriani Affenday, Ali Mamat**. "Named Entity Recognition Approaches ",IJCSNS, 2008

[3] **Carlo Meghini, Martin Doerr, Nicolas Spyratos**. "Sharing co-reference knowledge for data integration", pages 224-244 2008 EJC

[4] **N. Wacholder, R. Yael, C. Misook** , "Disambiguation of Proper Names in Text", Proceedings of the 5th Applied Natural Language Processing Conference, 1997

[5] **Liang-Jyh Wang , Wei-Chuan Li, Chao-Huang Chang**. "Recognizing Unregistered Names for Mandarin Word Identification". In Proc. International Conference on Computational Linguistics. 1992

[6] **Georgios Petasis, Frantz Vichot, Francis Wolinski, Georgios Paliouras, Vangelis Karkaletsis, Constantine D. Spyropoulos**. "Using Machine Learning to Maintain Rule-based Named-Entity Recognition and Classification Systems". In Proc. Conference of Association for Computational Linguistics. 2001.

[7] **Sotiris Boutsis, Iason Demiros, Voula Giouli, Maria Liakata, Harris Papageorgiou, Stelios Piperidis** . "A System for Recognition of Named Entities in Greek". In Proc. International Conference on Natural Language Processing. 2000.

[8] **William J. Black, Fabio Rinaldi, David Mowatt**. "Description of the NE System used for Muc-7". In Proc. Message Understanding Conference. 1998

[9] **Casey Whitelaw, J. Patrick.** , "Evaluating Corpora for Named Entity Recognition Using Character-Level Features". In Proc. Australian Conference on Artificial Intelligence. 2003

[10] **Joaquim F. Ferreira Da Silva , Zornitsa Kozareva , José Gabriel , Pereira Lopes**. "Cluster Analysis and Classification of Named Entities". In Proc. Conference on Language Resources and Evaluation.2004

[11] **Eckhard Bick**. "A Named Entity Recognizer for Danish". In Proc. Conference on Language Resources and Evaluation.2004

[12] **Alessandro Cucchiarelli, Paola Velardi**. "Unsupervised Named Entity Recognition Using Syntactic and Semantic Contextual Evidence". Computational Linguistics pages123-131, Cambridge: MIT Press 2001

[13] **Borislav Popov, A Kirilov, D. Maynard, D. Manov**. "Creation of reusable components and language resources for Named Entity Recognition in Russian". In Proc. Conference on Language Resources and Evaluation. 2004.

[14] **Dimitrios Kokkinakis**. "AVENTINUS, GATE and Swedish Lingware" . In Proc. of Nordic Computational Linguistics Conference.1998

[15] **Silviu Cucerzan,, D. Yarowsky**. "Language Independent Named Entity Recognition Combining Morphological and Contextual Evidence". In *Proc. Joint Sigdat Conference on Empirical Methods in Natural Language Processing and Very Large Corpora* 1999

[16] **Sam Coates-Stephens**. "The Analysis and Acquisition of Proper Names for the Understanding of Free Text". Computers and the Humanities 26.441-456, San Francisco:Morgan Kaufmann Publishers. 1992

[17] **Michael Fleischman**. "Automated Subcategorization of Named Entities". In Proc.Conference of the European Chapter of Association for Computational Linguistic,2001

[18] **Dan Shen, Jie Zhang, G. Zhou, Jian Su, C. L .Tan**. "Effective Adaptation of a Hidden Markov Model-based Named Entity Recognizer for Biomedical Domain". In Proc. Conference of Association for Computational Linguistics. Natural Language Processing in Biomedicine,2003

[19] **Y.C. Wu, T.K. Fan, Y.S. Lee, S.J Yen**, "Extracting Named Entities Using Support Vector Machines", Spring-Verlag, Berlin Heidelberg, 2006.

[20] **I. Budi, S. Bressan**, "Association Rules Mining for Name Entity Recognition", Proceedings of the Fourth International Conference on Web Information Systems Engineering, 2003

[21] **D. Appelt, and et. al**., "SRI International FASTUS system MUC-6 test results and analysis", Proceedings of the MUC-6,NIST, Morgan-Kaufmann Publisher, Columbia, 1995

[22] **D. Appelt, and et. al.,** "FASTUS: A finite state processor for information extraction from real-world text", Proceedings of IJCAI, 1993.

[23] **L. Iwanska, M. Croll, T. Yoon, and M. Adams**, "Wayne state university: Description of the UNO processing system as used for MUC-6", In Proc. of the MUC-6, NIST, Morgan-Kaufmann Publishers, Columbia, 1995

[24] **R. Morgan, and et. al.**"University of durham: Description of the LOLITA system as used for MUC-6" In Proc of the MUC-6, NIST, Morgan-Kaufmann Publishers, Columbia,1995.

[25] **H .Glaser, T. Lewy, I. Millard,B. Dowling**, "On Coreference and the Semantic Web". Technical Report , Electronics & Computer Science, University of Southampton. 2007

[26] **R.Gaizauskas, P.Rodgers, H.Cunningham, and K.Humphreys**. GATE UseGuide,1996.

[27] http://en.wikipedia.org/wiki/JAPE_(linguistics)

[28] **L. Ratinov and D. Roth**. "Design Challenges and Misconceptions in Named Entity Recognition", CoNLL 2009

[29] **Nicholas Delmonico Rizzolo**, "An Introduction to Learning Based Java", 2007 pages 195-211 .http://www.linguateca.pt/HAREM/actas/Capitulo_11-MotaSantos2008.pdf

[30] http://en.wikipedia.org/wiki/Learning_Based_Java

[31] **Jenny Rose Finkel, Trond Grenager, and Christopher Manning**. "Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling". Proceedings of the 43nd Annual Meeting of the Association for Computational Linguistics (ACL 2005), pages 363-370. 2005

[32] **Ai Kawazoe, Asanobu Kitamoto, Nigel Collier**, "Annotation of Coreference relation among linguistics and images in biological articles", In nats-www.informatik.uni-hamburg.de, 2004

[33] **Samuel Broscheit, Massimo Poesio, Simone Paolo Ponzetto, Kepa Joseba Rodr´ıguez, Lorenza Romano, Olga Uryupina, Yannick Versley, and Roberto Zanoli**. "BART: A multilingual anaphora resolution system". In Proceedings of SemEval-2. 2010

[34] **Thomas S. Morton Jeremy LaCivita** "WordFreak :An open tool for linguistic annotation" In HLT-NAACL ,2003

[35] **Rick Bennet, Christina Hengel Dittrich,Edward T. O' Neil, Barbara B.Tillet** 'VIAF (Virtual International Authority File): Linking Die Deutsche Bibliothek and Library of Congress Name Authority Files, World library and information Congress: 72ND IFLA general conference and council,2006

[36] LEAF Project, http://www.leaf-eu.org Accessed 26 November 2010

[37] **Patrice Landry**, " Providing multilingual subject access through linking of subject heading languages: The MACS approach", Workshop on Advanced Technologies for Digital Libraries, 2009

[38] **Library of Congress Classification Outline**, Library of Congress, http://www.loc.gov/catdir/cpso/lcco/, Accessed 26 November 2010

[39] **Andreas Hotho, Robert Jäschke, Christoph Schmitz, Gerd Stumme** "Trend Detection in Folksonomies". Pages 56-80 SAMT 2006

[40] **Jennifer Trant**, "Studying Social Tagging and Folksonomy: A Review and Framework". Journal of Digital Information 2009

[41] **Nick Crofts, Martin Doerr, Tony Gill, Stephen Stead, Matthew Stiff** , "Definition of the CIDOC Conceptual Reference Model"**,** http://www.cidoc-crm.org/docs/cidoc_crm_version_5.0.2.pdf, January 2010

[42] **H. Halpin and P. J. Hayes**. "When owl:sameAs isn't the same: An analysis of identity links on the SemanticWeb". In Proceedings of the WWW2010 workshop on Linked Data on the Web,LDOW2010, 2010

Appendix

Appendix A.1 RDF Source

```
<rdf:RDF
        xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#"
         xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
        xmlns:xsd="http://www.w3.org/2001/XMLSchema#">
<rdfs:Class rdf:ID="CR_Named_Entity">
</rdfs:Class>
<rdfs:Class rdf:ID="E21_Person">
        <rdfs:subClassOf rdf:resource="#CR_Named_Entity"/>
</rdfs:Class>
<rdfs:Class rdf:ID="E53_Place">
        <rdfs:subClassOf rdf:resource="#CR_Named_Entity"/>
</rdfs:Class>
<rdfs:Class rdf:ID="E5_Event">
        <rdfs:subClassOf rdf:resource="#CR_Named_Entity"/>
</rdfs:Class>
<rdfs:Class rdf:ID="E67_Birth">
        <rdfs:subClassOf rdf:resource="#E5_Event"/>
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Meet">
        <rdfs:subClassOf rdf:resource="#E5_Event"/>
</rdfs:Class>
<rdfs:Class rdf:ID="E69_Death">
        <rdfs:subClassOf rdf:resource="#E5_Event"/>
</rdfs:Class>
```

<rdfs:Class rdf:ID="CR_URL">

</rdfs:Class>

<rdfs:Class rdf:ID="E73_Informational_Object">

</rdfs:Class>

<rdfs:Class rdf:ID="CR_HTML_Document">

<rdfs:subClassOf rdf:resource="#E73_Informational_Object"/>

</rdfs:Class>

<rdfs:Class rdf:ID="CR_Digital_Image">

<rdfs:subClassOf rdf:resource="#E73_Informational_Object"/>

</rdfs:Class>

<rdfs:Class rdf:ID="CR_PDF_Document">

<rdfs:subClassOf rdf:resource="#E73_Informational_Object"/>

</rdfs:Class>

<rdfs:Class rdf:ID="CR_Area">

</rdfs:Class>

<rdfs:Class rdf:ID="CR_HTML_Area">

<rdfs:subClassOf rdf:resource="#CR_Area"/>

</rdfs:Class>

<rdfs:Class rdf:ID="CR_2D_Area">

<rdfs:subClassOf rdf:resource="#CR_Area"/>

</rdfs:Class>

<rdfs:Class rdf:ID="CR_PDF_Area">

<rdfs:subClassOf rdf:resource="#CR_Area"/>

</rdfs:Class>

<rdfs:Class rdf:ID="CR_Citation">

</rdfs:Class>

<rdfs:Class rdf:ID="E39_Actor">

</rdfs:Class>

<rdfs:Class rdf:ID="E42_Identifier">

</rdfs:Class>

```
<rdfs:Class rdf:ID="E52_Time_Span">
</rdfs:Class>
<rdfs:Class rdf:ID="E14_Time_Primitive">
</rdfs:Class>
<rdfs:Class rdf:ID="CR_State">
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Role">
        <rdfs:subClassOf rdf:resource="#CR_State"/>
</rdfs:Class>
<rdfs:Class rdf:ID="CR_E74_Group">
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Identity_Assumption">
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Person_Identity_Assumption">
        <rdfs:subClassOf rdf:resource="#CR_Identity_Assumption"/>
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Place_Identity_Assumption">
        <rdfs:subClassOf rdf:resource="#CR_Identity_Assumption"/>
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Geopolitical_subordination">
        <rdfs:subClassOf rdf:resource="#CR_State"/>
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Longitude">
</rdfs:Class>
<rdfs:Class rdf:ID="CR_Latitude">
</rdfs:Class>
<rdf:Property rdf:ID="CR_Is_Locate_on">
        <rdfs:domain rdf:resource="#E73_Informational_Object"/>
        <rdfs:range rdf:resource="#CR_URL"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="CR_Date_Introduced">
        <rdfs:domain rdf:resource="#E73_Informational_Object"/>
        <rdfs:range rdf:resource="xsd:dateTime"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_created_on">
        <rdfs:domain rdf:resource="#CR_HTML_Area"/>
        <rdfs:range rdf:resource="xsd:dateTime"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_title">
        <rdfs:domain rdf:resource="#CR_HTML_Document"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_Parent_Page">
        <rdfs:domain rdf:resource="#CR_Digital_Image"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_PDF_has_Author">
        <rdfs:domain rdf:resource="#CR_PDF_Document"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_PDF_has_Title">
        <rdfs:domain rdf:resource="#CR_PDF_Document"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_included_in">
        <rdfs:domain rdf:resource="#CR_Area"/>
        <rdfs:range rdf:resource="#E73_Informational_Object"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_determines_object">
        <rdfs:domain rdf:resource="#CR_Citation"/>
```

```
        <rdfs:range rdf:resource="#E73_Informational_Object"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_refers_to">
        <rdfs:domain rdf:resource="#E73_Informational_Object"/>
        <rdfs:range rdf:resource="#CR_Named_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_included_in_PDF_page">
        <rdfs:domain rdf:resource="#CR_PDF_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
        <rdfs:subPropertyOf rdf:resource="#CR_included_in"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_included_in_HTML_Page">
        <rdfs:domain rdf:resource="#CR_HTML_Area"/>
        <rdfs:range rdf:resource="#CR_HTML_Document"/>
        <rdfs:subPropertyOf rdf:resource="#CR_included_in"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_included_in_Digital_Image">
        <rdfs:domain rdf:resource="#CR_2D_Area"/>
        <rdfs:range rdf:resource="#CR_Digital_Image"/>
        <rdfs:subPropertyOf rdf:resource="#CR_included_in"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_Area_Start">
        <rdfs:domain rdf:resource="#CR_HTML_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_Area_End">
        <rdfs:domain rdf:resource="#CR_HTML_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="CR_has_Text">
        <rdfs:domain rdf:resource="#CR_HTML_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_down_right_point">
        <rdfs:domain rdf:resource="#CR_2D_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_PDF_has_up_left_point">
        <rdfs:domain rdf:resource="#CR_PDF_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_PDF_has_down_right_point">
        <rdfs:domain rdf:resource="#CR_PDF_Area"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_supported_by">
        <rdfs:domain rdf:resource="#CR_Citation"/>
        <rdfs:range rdf:resource="#E39_Actor"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_cites">
        <rdfs:domain rdf:resource="#CR_Citation"/>
        <rdfs:range rdf:resource="#CR_Named_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_introduced_on">
        <rdfs:domain rdf:resource="#CR_Citation"/>
        <rdfs:range rdf:resource="xsd:dateTime"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_password">
        <rdfs:domain rdf:resource="#E39_Actor"/>
```

```
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_Name">
        <rdfs:domain rdf:resource="#CR_Named_Entity"/>
        <rdfs:range rdf:resource="http://www.w3.org/2000/01/rdf-schema#Literal"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_Known_by">
        <rdfs:domain rdf:resource="#CR_Named_Entity"/>
        <rdfs:range rdf:resource="#E39_Actor"/>
</rdf:Property>
<rdf:Property rdf:ID="P48_has_preferred_identifier">
        <rdfs:domain rdf:resource="#CR_Named_Entity"/>
        <rdfs:range rdf:resource="#E42_Identifier"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_this_Entity">
        <rdfs:domain rdf:resource="#CR_Identity_Assumption"/>
        <rdfs:range rdf:resource="#CR_Named_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_same_with_Entity">
        <rdfs:domain rdf:resource="#CR_Identity_Assumption"/>
        <rdfs:range rdf:resource="#CR_Named_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_not_same_with_Entity">
        <rdfs:domain rdf:resource="#CR_Identity_Assumption"/>
        <rdfs:range rdf:resource="#CR_Named_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_possible_same_with_Entity">
        <rdfs:domain rdf:resource="#CR_Identity_Assumption"/>
        <rdfs:range rdf:resource="#CR_Named_Entity"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="CR_this_Person">
        <rdfs:domain rdf:resource="#CR_Person_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_this_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_same_with_Person">
        <rdfs:domain rdf:resource="#CR_Person_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_is_same_with_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_not_same_with_Person">
        <rdfs:domain rdf:resource="#CR_Person_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_is_not_same_with_Entity"/>
</rdf:Property>

<rdf:Property rdf:ID="CR_is_possible_same_with_Person">
        <rdfs:domain rdf:resource="#CR_Person_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_is_possible_same_with_Entity"/>
</rdf:Property>

<rdf:Property rdf:ID="CR_this_Place">
        <rdfs:domain rdf:resource="#CR_Place_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E53_Place"/>
        <rdfs:subPropertyOf rdf:resource="#CR_this_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_same_with_Place">
        <rdfs:domain rdf:resource="#CR_Place_Identity_Assumption"/>
```

```
        <rdfs:range rdf:resource="#E53_Place"/>
        <rdfs:subPropertyOf rdf:resource="#CR_is_same_with_Entity"/>
</rdf:Property>

<rdf:Property rdf:ID="CR_is_not_same_with_Place">
        <rdfs:domain rdf:resource="#CR_Place_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E53_Place"/>
        <rdfs:subPropertyOf rdf:resource="#CR_is_not_same_with_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_is_possible_same_with_Place">
        <rdfs:domain rdf:resource="#CR_Place_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E53_Place"/>
        <rdfs:subPropertyOf rdf:resource="#CR_is_possible_same_with_Entity"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_longitude">
        <rdfs:domain rdf:resource="#E53_Place"/>
        <rdfs:range rdf:resource="#CR_Longitude"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_latitude">
        <rdfs:domain rdf:resource="#E53_Place"/>
        <rdfs:range rdf:resource="#CR_Latitude"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_in_state">
        <rdfs:domain rdf:resource="#E53_Place"/>
        <rdfs:range rdf:resource="#CR_Geopolitical_Subordination"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_geopolitical_subordination">
        <rdfs:domain rdf:resource="#CR_Geopolitical_Subordination"/>
        <rdfs:range rdf:resource="#E53_Place"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="P89_was_Born">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E67_Birth"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_get_in">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#CR_Meet"/>
</rdf:Property>
<rdf:Property rdf:ID="P100_Died_in">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E69_Death"/>
</rdf:Property>
<rdf:Property rdf:ID="P4_has_time_span">
        <rdfs:domain rdf:resource="#E5_Event"/>
        <rdfs:range rdf:resource="#E52_Time_Span"/>
</rdf:Property>
<rdf:Property rdf:ID="P81_ongoing_throuh_out">
        <rdfs:domain rdf:resource="#E52_Time_Span"/>
        <rdfs:range rdf:resource="#E14_Time_Primitive"/>
</rdf:Property>
<rdf:Property rdf:ID="P82_at_some_time_within">
        <rdfs:domain rdf:resource="#E52_Time_Span"/>
        <rdfs:range rdf:resource="#E14_Time_Primitive"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_starts">
        <rdfs:domain rdf:resource="#CR_State"/>
        <rdfs:range rdf:resource="#E52_Time_Span"/>
</rdf:Property>
```

```
<rdf:Property rdf:ID="CR_finishes">
        <rdfs:domain rdf:resource="#CR_State"/>
        <rdfs:range rdf:resource="#E52_Time_Span"/>
</rdf:Property>

<rdf:Property rdf:ID="P107_is_current_or_former_member_of">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E74_Group"/>
</rdf:Property>

<rdf:Property rdf:ID="has_role">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="# CR_Role "/>
</rdf:Property>

<rdf:Property rdf:ID="CR_in_group">
        <rdfs:domain rdf:resource="# CR_Role "/>
        <rdfs:range rdf:resource="#E74_Group"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_has_created_by">
        <rdfs:domain rdf:resource="#CR_Identity_Assumption"/>
        <rdfs:range rdf:resource="#E39_Actor"/>
</rdf:Property>

<rdf:Property rdf:ID="CR_social_relation">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
</rdf:Property>

<rdf:Property rdf:ID="CR_parent_of">
```

```
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_child_of">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_spouse_of">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_grandchild_of">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_grandparent_of">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_cousin_of">
        <rdfs:domain rdf:resource="#E21_Person"/>
        <rdfs:range rdf:resource="#E21_Person"/>
        <rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>
</rdf:Property>
<rdf:Property rdf:ID="CR_uncle_of">
```

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

<rdf:Property rdf:ID="CR_aunt_of">

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

<rdf:Property rdf:ID="CR_niece_of">

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

<rdf:Property rdf:ID="CR_nephew_of">

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

<rdf:Property rdf:ID="CR_brother_of">

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

<rdf:Property rdf:ID="CR_sister_of">

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

<rdf:Property rdf:ID="CR_other_relation">

<rdfs:domain rdf:resource="#E21_Person"/>

<rdfs:range rdf:resource="#E21_Person"/>

<rdfs:subPropertyOf rdf:resource="#CR_social_relation"/>

</rdf:Property>

</rdf:RDF>